
INVESTIGACION DE METODOLOGIAS Y TECNICAS PARA LA GENERACION, REPRESENTACION Y VISUALIZACION DE OBJETOS TRIDIMENSIONALES A PARTIR DE IMAGENES TOMOGRAFICAS

*Trabajo final presentado por Luis. R. Blando y Diego A. Viñas a la Escuela
de Ingeniería de Sistemas de la Universidad Católica de Córdoba
como requisito final para la obtención del
título de Ingenieros de Sistemas.*

Asesor: Ing. Juan A. Ronda

Escuela de Ingeniería de Sistemas

UNIVERSIDAD CATOLICA DE CORDOBA

Córdoba, 28 de Septiembre de 1992

Agradecimientos

Deseamos agradecer a: Gerardo Fisanotti, de Galmes y Casale SRL por darnos la "inspiración" para llevar a cabo este trabajo; a la gente del FAMAF, sin cuya bibliografía nunca hubieramos iniciado la investigación; al Centro de Tomografía Computada Córdoba, por permitimos investigar en sus equipos; al Dr. Eduardo Noguera, del Hospital Privado Córdoba, por facilitarnos las placas utilizadas como ejemplos; a Marcelo "Condor" Cortéz, por facilitarnos el scanner; a los Dres. Gabor T. Herman, William Barrett y J. K. Udupa, por su colaboración con información; a Federico Mendoza, de PC Shop SRL por incontables páginas impresas; a la UCC; a Diego Linares por su colaboración con Windows; a Manuel Viale por hacer las veces de correo; a Marina Sanchez Flores y Dante Ramos por sus comentarios y críticas y a Marcelo De Stefano por su exhaustivo análisis y sus consejos para mejorar este trabajo.

Muchas gracias Ing. Juan Ronda por haber mantenido el rumbo de este trabajo y habernos ayudado con la bibliografía a pesar de los múltiples contratiempos sufridos.

Gracias a toda mi familia, que soportó siempre ser relegada a un segundo plano; a mi Tía, que hizo las veces de madre; a mis padres, que siempre se jugaron por mí; a Diego, por la amistad y la paciencia; a Laura, por siempre estar. Gracias Nona, por traerme hasta aquí.

Luis

Quiero agradecer a Cecilia por su infinita paciencia y comprensión en mis innumerables "ausencias" y a Gonzalo que soportó las continuas transformaciones de su cuarto en un improvisado centro de cómputos. Un agradecimiento muy especial a Luis por su inalterable alma de diamante. Finalmente, quiero dedicar este trabajo a mis padres por su inmenso amor y su constante apoyo a lo largo de toda mi carrera.

Diego

Córdoba, 28 de Septiembre de 1992

Contenido

INTRODUCCION	5
§ 1.1 Resumen	5
§ 1.2 Justificación	5
§ 1.3 Objetivos del trabajo	6
TOMOGRAFIA COMPUTADA	7
§ 2.1 Conceptos generales sobre tomografía	7
§ 2.2 Tomógrafo Axial Computado	8
§ 2.3 Aplicaciones del TAC en el campo de la medicina	9
§ 2.4 Descripción de la Situación Actual	10
§ 2.5 Diseño general del sistema	10
GENERACION	12
§ 3.1 Adquisición de Datos	12
3.1.1 Salida Numérica via Drive	12
3.1.2 Digitalización Directa de la Pantalla del Tomógrafo	13
3.1.3 Digitalización de las Placas Mediante Scanner	14
§ 3.2 Representación de las Imágenes 2D	15
Que queremos decir con "registración"?	16
3.2.1 Identificación del Punto en Común (Punto Pivote)	17
3.2.2 Determinación de la Región de Interés (ROI)	18
3.2.3 Determinación del Volumen de Interés (VOI)	18
§ 3.3 Transformaciones de la Imagen	19
3.3.1 Transformada de Fourier	19
3.3.1.1 Separabilidad	20
3.3.1.2 Translación	21
3.3.1.3 Periodicidad y Simetría Conjugada	22
3.3.1.4 Rotación	22
3.3.1.5 Distribución y Escalamiento (Linearidad)	23
3.3.1.6 Laplaciano	23
3.3.1.7 Convolución y Correlación	23
3.3.2 Transformada Discreta del Coseno	24
§ 3.4 Mejoramiento de la Imagen	25
3.4.1 Métodos en el Dominio de Frecuencia	25
3.4.2 Métodos en el Dominio Espacial	26
3.4.2.1 Filtro de la Media	27
3.4.2.2 Filtro de la Mediana	27

REPRESENTACION	29
§ 4.1 Obtención de Voxels Cúbicos	29
4.1.1 Interpolación Lineal	31
4.1.2 Espacio de Almacenamiento de Datos	33
$A = ResX * ResY (N + (N - 1) * CInterp)$	33
§ 4.2 Identificación del órgano de interés	34
4.2.1 Atlas	35
4.2.2 Segmentación Interactiva	35
4.2.3 Threshold (Filtro de Umbral)	36
§ 4.3 Estructuras de Datos	37
4.3.1 Volume Rendering	38
4.3.2 Surface Rendering	38
4.3.2.1 Contornos Dirigidos Nivelados	39
4.3.2.2 Parches Triangulares	41
4.3.2.2.1 Algoritmo de Decisión de Encadenamiento	41
4.3.2.2.2 Single Branching entre Contornos Convexos	43
4.3.2.2.3 Obtención de la Cápsula Convexa (Convex-Hull)	48
4.3.2.2.4 Secuencia Final de Pasos	49
4.3.2.2.5 Multiple Branching	50
VISUALIZACION	54
§ 5.1 Introducción	54
§ 5.2 Volume Rendering	54
5.2.1 Método Back-To-Front para Visualización de Volúmenes	55
5.2.2 Método Híbrido	58
§ 5.3 Surface Rendering	59
5.3.1 Contornos Dirigidos Nivelados	59
5.3.2 Parches Triangulares	60
§ 5.4 Sombreado (Shading)	60
5.4.1 Nuestra Implementación	61
§ 5.5 Entorno GUI (Graphical User Interface)	62
5.5.1 Modalidades de Visualización	63
DETALLES TECNICOS	65
§ 6.1 Equipamiento Utilizado	65
§ 6.2 Módulos Componentes del Sistema	65
§ 6.3 Detalles de Software	66
CONCLUSIONES Y RECOMENDACIONES	68
§ 7.1 Conclusiones Sobre el Trabajo	68
§ 7.2 Recomendaciones Para Futuros Trabajos	69
GLOSARIO	70
BIBLIOGRAFIA	73
APENDICE	76

CAPITULO I

INTRODUCCION

§ 1.1 Resumen

El presente trabajo tiene por objeto presentar una metodología adecuada para la representación y visualización de objetos tridimensionales generados a partir de cortes transversales igualmente espaciados de un objeto.

Se presenta como modelo la reconstrucción de la estructura ósea de algunas partes del cuerpo a partir de placas obtenidas mediante estudios de tomografía axial computada.

Dado que el diseño e implementación de un producto comercial está fuera del alcance de este trabajo, el objetivo es investigar los distintos métodos y algoritmos disponibles, proponiendo una metodología para todo el proceso.

§ 1.2 Justificación

La reconstrucción de objetos tridimensionales es un área en constante avance e investigación. Debido a la disminución progresiva de los precios de hardware y software, tecnologías (y sus aplicaciones), que en otras épocas hubieran parecido increíbles, son accesibles hoy para un modesto presupuesto y están disponibles para computadoras de tipo personal. Esto implica, entre otras cosas, la transición de 2D a 3D sobre plataformas desktop.

Con esta "revolución 3D" se crea la necesidad de proveer datos en tres dimensiones para todos los elementos estudiados. Lejos de esta situación, la mayoría de los métodos de adquisición utilizados en la industria/ingeniería -áreas comúnmente vinculadas a necesidades de modelado 3D- siguen generando información bidimensional.

Durante los 80, toda la "computación gráfica" (o de modelado de elementos, como ser CAD) tridimensional estaba restringida, debido a los altos costos computacionales, a estaciones de trabajo especialmente diseñadas para esta tarea, mientras que los equipos personales se limitaban a tareas simples en dos dimensiones.

Con el consistente aumento en la capacidad de los equipos PC, a principios de los 90 se los empieza a utilizar en el modelado tridimensional. Se hace necesario, entonces, reconstruir en tres dimensiones los objetos de la realidad, a partir de datos bidimensionales.

Un ejemplo típico es el equipo de Tomografía Axial Computada (TAC, de ahora en más), que va recorriendo el cuerpo obteniendo cortes axiales y generando una imagen de cada corte realizado. Los equipos de TAC realizan cortes coplanares con gran precisión. Se observó así la posibilidad de reconstruir un órgano determinado a partir de una sucesión de cortes.

Si bien los tomógrafos modernos disponen de software para reconstruir en 3D los órganos en sus consolas, esta información se pierde en el transporte al médico, ya que este sólo recibe una o varias placas, que, aunque quizás con una vista en 3D del órgano en cuestión, no le permiten interactuar con la información ni modificar el punto de vista.

Dado que la información volumétrica completa esta guardada en el equipo TAC, el problema está en el transporte de esta información. La propuesta es, entonces, diseñar un método para permitirle a un médico, con tan solo un equipo PC, interactuar con la información entregada por el equipo TAC, previamente procesada en centro de diagnóstico por imágenes.

§ 1.3 Objetivos del trabajo

Se pretende definir una metodología para el procesamiento de imágenes bidimensionales y la reconstrucción y visualización de objetos tridimensionales, con algunas restricciones:

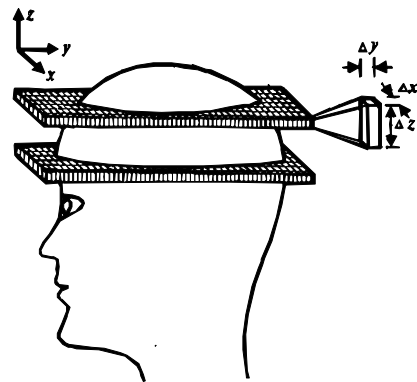
1. La visualización deberá realizarse enteramente en un equipo tipo PC
2. Se deberá diseñar algún método y optimizar los algoritmos para permitir, de manera interactiva, modificar los parámetros de visualización sin producir tiempos de espera demasiado largos en la generación de la imagen
3. El transporte de información del centro de diagnóstico por imágenes al médico se hará por medios magnéticos, debiéndose optimizar el espacio de datos dada la limitada capacidad de los diskettes utilizados comúnmente en los equipos PC.

CAPITULO II

TOMOGRAFIA COMPUTADA

§ 2.1 Conceptos generales sobre tomografía

La tomografía es básicamente una colección de cortes planos transversales, cada uno de los cuales se obtiene por reconstrucción a partir de la medición de la radiación absorbida cuando se ilumina al cuerpo con un haz de rayos contenidos en ese plano, y desde varios ángulos alrededor del eje.



• Ilustración 1-A: Ubicación espacial de los cortes.
Relación entre resoluciones en X,Y y Z.

La imagen tomográfica se forma iluminando el objeto con alguna forma de energía (rayos-X, microondas o ultrasonido) y midiendo la cantidad de la misma que atraviesa el objeto. En la (Ilustración 1-B) se observa una imagen obtenida al irradiar un cráneo a una altura determinada.

En el caso de los Rayos-X, la medida puede ser la amplitud o el tiempo de arribo de la señal recibida. De esta forma se obtiene una estimación de un coeficiente de atenuación o índice de refracción. Aún cuando el rayo no atravesase el objeto, es posible, a menudo, usar técnicas algebraicas para conformar la imagen.

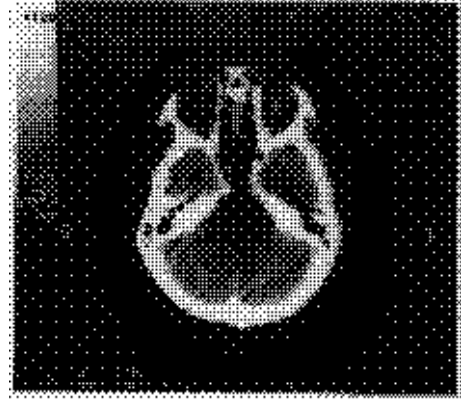
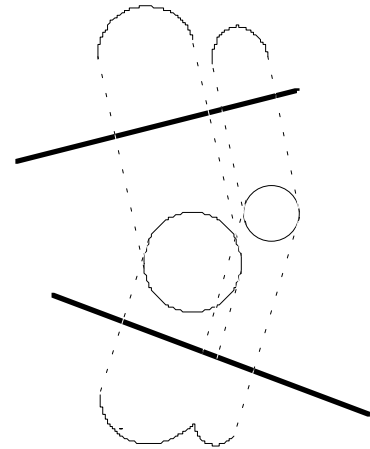


Ilustración 1-B

Fundamentalmente, la placa tomográfica se obtiene reconstruyendo una imagen a partir de sus proyecciones. En el sentido estricto de la palabra, una proyección en un determinado ángulo, es la integral de la densidad de la imagen en la dirección especificada por el ángulo (Figura 1). Por otro lado, en un sentido más amplio, proyección se refiere a la información derivada de la energía transmitida, cuando un objeto es irradiado desde un ángulo en particular; la frase "proyección defractada" es oportuna para ser usada cuando la fuente de energía es defractante, como en el caso de ultrasonido y microondas.

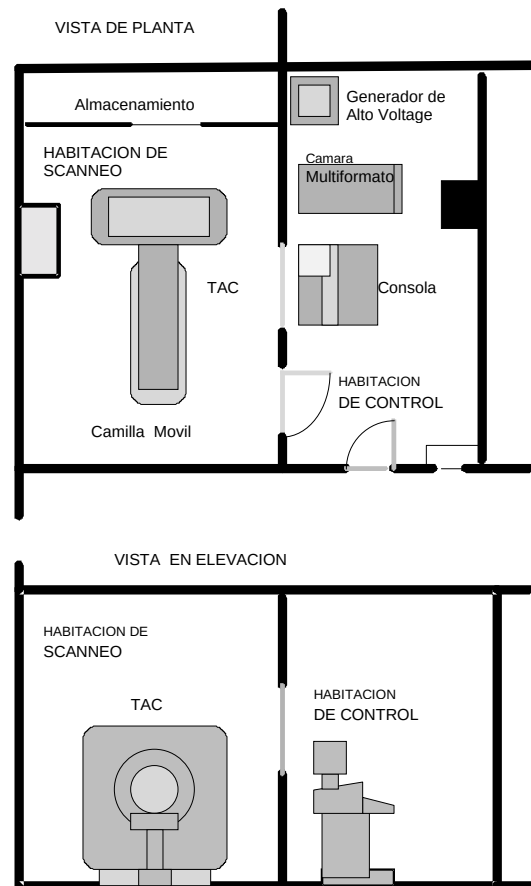


• Fig. 1: Dos proyecciones de un objeto consistente en un par de cilindros

§ 2.2 Tomógrafo Axial Computado

El equipo que realiza la irradiación se denomina Tomógrafo Axial Computado (TAC) y consiste en un dispositivo por donde se desplaza la camilla con el paciente al que se le ha de realizar el estudio. Dicho dispositivo, que es el encargado de realizar la emisión de Rayos-X sobre el órgano de interés, se encuentra en una habitación aislada, debido a que la exposición frecuente (del operador del equipo) a los rayos es nociva para el cuerpo humano.

Por lo tanto, la consola de control, como así también el computador del equipo, se hallan en una habitación contigua al dispositivo de scaneo. Es allí donde el técnico opera el equipo y selecciona los distintos cortes para un estudio determinado.



• Ilustración 2: Layout de un sistema de tomografía computada

La (Ilustración 2) nos muestra la disposición física (layout) de un sistema de tomografía computada en un hipotético centro de tomografía computada.

§ 2.3 Aplicaciones del TAC en el campo de la medicina

El impacto de esta técnica en el diagnóstico médico ha sido revolucionario, ya que posibilita un estudio profundo y preciso de órganos humanos internos con un alto grado de seguridad para el paciente.

El rango de aplicación del diagnóstico por imágenes en medicina se ha extendido considerablemente en los últimos años. Así, por ejemplo, los sistemas TAC son en la actualidad muy utilizados en neurología, ortopedia, otorinolaringología, etc., como herramienta fundamental de diagnóstico y soporte al profesional médico.

§ 2.4 Descripción de la Situación Actual

En la configuración básica de todo computador de TAC viene incorporado un disco rígido -donde es posible almacenar un número muy limitado de estudios debido a su baja capacidad- y un drive de disco flexible.

Ninguno de los dos dispositivos de almacenamiento son aprovechados suficientemente debido a que los resultados de los estudios son impresos en una placa y borrados del disco posteriormente.

El drive de disco flexible, en general, no es utilizado en absoluto pues solo sirve para transportar la información acerca de un estudio determinado hacia otro centro de tomografía, cuando este último está equipado con el mismo modelo de TAC (o uno compatible).

Por todo ello, si bien los sistemas de TAC son herramientas muy poderosas para asistir en el diagnóstico, mucha de la capacidad potencial de estos aparatos para transmitir información se ve subutilizada debido al medio de "transporte" (placas) desde los centros de diagnóstico por imágenes al médico especialista.

Este transporte de información consiste a menudo en una (o más) placa(s) con imágenes del paciente, frecuentemente seleccionadas por el especialista en el centro de diagnóstico. Con esta metodología se está perdiendo gran cantidad de información, especialmente volumétrica, que podría ser utilizada tanto por el responsable del aparato de TAC como también directamente por el médico que solicitó el estudio.

Sería deseable, entonces, alguna solución tecnológica que "transporte" toda la información obtenida por el tomógrafo a las manos del usuario.

Este trabajo expone nuestras experiencias en la investigación de distintos métodos para la realización de un sistema de este tipo.

§ 2.5 Diseño general del sistema

El punto de partida del trabajo son las imágenes bidimensionales generadas por el aparato de TAC. El de llegada es un sistema en el cual el usuario pueda visualizar interactivamente algún órgano específico (esto es, modificar las condiciones del entorno - punto de vista, iluminación, etc.- y recibir inmediata respuesta).

Para ofrecer la mayor cantidad de información posible es necesario pasar de una representación bidimensional a una tridimensional, relacionando adecuadamente las varias imágenes bidimensionales obtenidas del tomógrafo. Una vez generado este "volumen de información" es posible estudiarlo con mucha mayor flexibilidad mediante el uso de una computadora personal que a través de la observación de la(s) placa(s) utilizadas comúnmente.

Para lograr este objetivo, se debe primero seleccionar una manera de obtener los datos bidimensionales. Posteriormente es conveniente diseñar una metodología apropiada para representar y procesar la imagen digital. Es aconsejable un filtrado de estos datos para eliminar el ruido realizando, si es necesario, un realce de bordes y aumento de contraste, de modo de poder aislar convenientemente el objeto de interés.

Luego se debería formar un espacio tridimensional de datos a partir de un conjunto de cortes bidimensionales. Obtenido este, se continúa identificando el objeto (órgano) de interés (por ejemplo, hueso). A continuación se debe representar este objeto tridimensional mediante alguna forma adecuada para obtener una visualización coherente. Por último, se debe diseñar un sistema para usuario final (end-user system) que capte y almacene la información generada por los pasos anteriores y la presente al usuario de forma amena y simple.

Es conveniente explicar el presente trabajo de la forma en la que fué realizado, para lo cual es posible diferenciar, con fines puramente didácticos, las siguientes etapas en el desarrollo:

- **GENERACION**

- Adquisición de los datos
- Representación de Imágenes en 2D
- Procesamiento Digital de Imágenes en 2D

- **REPRESENTACION**

- Obtención de Voxels Cúbicos
- Identificación de Órganos
- Estructuras de Datos

- **VISUALIZACIÓN**

- Entorno GUI
- Sistema Gráfico en 3D
- Wireframe Representation
- Sombreado / Coloreado (Shading)

CAPITULO III

GENERACION

Esta etapa comprende el proceso de obtención de la información del tomógrafo computado, su representación en el espacio bidimensional y el procesamiento digital de estas imágenes de manera de proveer el material apropiado para una representación en tres dimensiones lo más fidedigna posible.

§ 3.1 Adquisición de Datos

El problema de adquirir los datos desde el tomógrafo fue uno de los mayores obstáculos en la realización de este sistema. Antes de hacer un estudio detallado de las distintas alternativas existentes es importante notar que la selección de un método de adquisición sobre otro condiciona en gran medida el trabajo posterior que se deberá realizar con las imágenes. Es así que hemos ordenado los distintos métodos de adquisición en según su "deseabilidad".

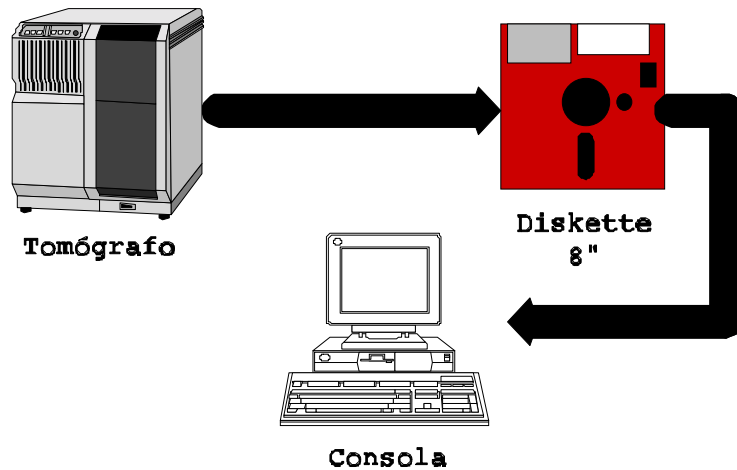
Básicamente, existen tres formas posibles de extraer la información del equipo TAC:

1. Salida numérica via drive
2. Digitalización directa de la pantalla del tomógrafo
3. Digitalización de las placas mediante scanner

3.1.1 Salida Numérica via Drive

Consiste en grabar en diskette, la información numérica que define las imágenes obtenidas en el estudio, previo a su borrado del disco rígido del equipo TAC (Ilustración 3).

Este método de transporte es el mas deseado, ya que se consigue toda la información que maneja el tomógrafo (incluidos ciertos parámetros que sino deberán ser estimados) y no se introduce ruido en el proceso de adquisición.



• Ilustración 3: Flujo de información desde el tomógrafo hasta la PC

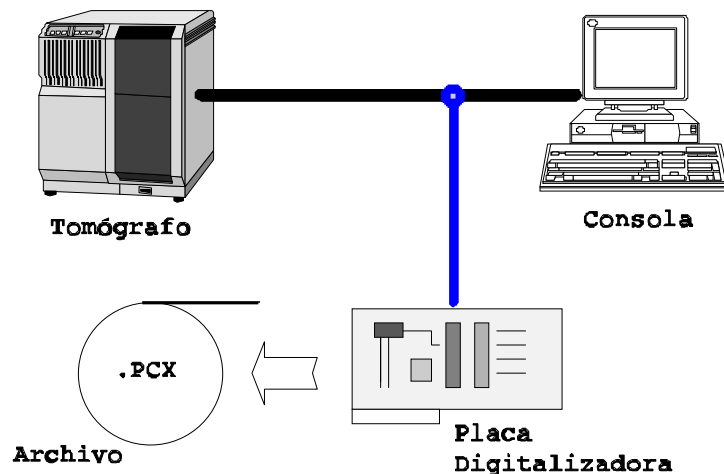
Para implementar este método es necesario disponer de:

- Un método de acceder a la información desde el dispositivo de almacenamiento del tomógrafo. Lo cual ya constituye un problema debido a que el medio de transporte externo que utilizan los TAC son normalmente diskettes de 8", que son poco comunes.
- El formato en que se encuentran almacenados los datos. Esto no es simple ya que las distintas marcas de fabricantes de tomógrafos tienen sus propios formatos "propietarios" de archivos y, en muchos casos, usan el encriptado como un medio para mantener el mercado. Se escribió a Toshiba-Japón y Siemens-Alemania para obtener los formatos, sin recibirse nunca la respuesta de ninguno de ellos.

3.1.2 Digitalización Directa de la Pantalla del Tomógrafo

Consiste en interceptar la señal de video que recibe el monitor del TAC y enviarla a la computadora personal a través de una placa digitalizadora (Ilustración 4).

Si bien este método es menos directo que el anterior, es muy deseable ya que, fundamentalmente, se mantiene la relación espacial entre los distintos cortes. Esto es, entre el corte $C(n)$ y el $C(n+1)$ existe una relación espacial de manera tal que $C(n+1)$ esta exactamente sobre (o bajo) $C(n)$.



• Ilustración 4: Flujo de datos para el método de adquisición por placa digitalizadora de video.

Como la imagen en la pantalla del tomógrafo esta siempre encuadrada de la misma manera (para un mismo estudio, naturalmente) esta coherencia espacial nos libera de la necesidad de encuadrar las distintas imágenes manualmente.

Además, los tomógrafos generalmente generan la imagen a intervalos regulares de tiempo, por lo que sería factible diseñar el sistema de reconocimiento para que funcione automáticamente, de manera sincronizada con el tomógrafo.

Como aspecto negativo se puede mencionar la introducción de ruido en el proceso de digitalización (la mayoría de los equipos TAC trabajan con una resolución de 12 bits, mientras que los digitalizadores comunes disponen de sólo 8).

Para implementar este sistema hace falta disponer de una placa de adquisición de datos que se adecue a las especificaciones técnicas del sistema de video del tomógrafo (que por lo general, no son compatibles con ninguno de los sistemas mas conocidos como NTSC, PAL o SECAM).

Luego, es imprescindible contar con los medios para construir la placa digitalizadora y montar la PC de modo que reciba la señal de video simultáneamente con el monitor del TAC.

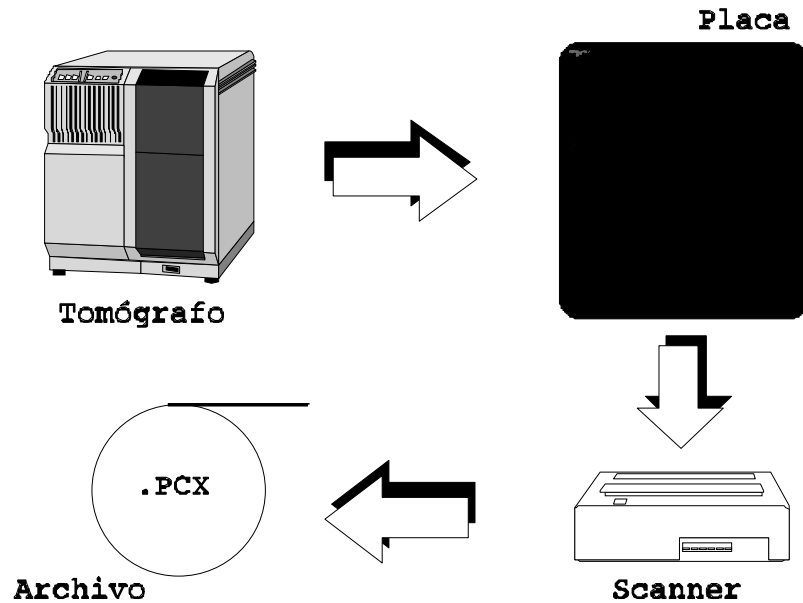
Es relativamente más fácil acceder a las especificaciones técnicas de la señal de video que obtener el formato propietario de grabación. Esto se debe al hecho de que, en nuestro país, el mantenimiento de estos equipos se realiza a nivel de hardware, y todo aquello que respecta al formato y método de almacenamiento, es desconocido por los técnicos de las empresas locales distribuidoras de TAC. No obstante, debido a los costos relativamente altos de estas placas no pudimos disponer de ellas. Utilizamos, entonces, el método descrito a continuación.

3.1.3 Digitalización de las Placas Mediante Scanner

Este método consiste en obtener, via scanner, las imágenes impresas en placas tomográficas (Ilustración 5).

Si bien este método es mucho mas artesanal y menos perfecto que los anteriores, utilizando un scanner de tonos de grises se puede obtener una buena calidad en la imagen.

Con este método se introducen una serie de errores de registración entre los distintos cortes. Estos son debidos a pequeñas diferencias en la orientación y ubicación de las placas en el momento del scaneo.



• Ilustración 5: Flujo de datos para el método de adquisición por digitalización de las placas con scanner

Estas diferencias pueden ser de traslación como así también de orientación. Mas adelante se explicará nuestra solución a estos problemas.

§ 3.2 Representación de las Imágenes 2D

Si utilizamos los sistemas de adquisición descritos en (3.1.1) o (3.1.2), es directa la traducción de la información a un formato manejable por nosotros.

En el caso de utilizar el método descrito en (3.1.1), se debería implementar un sistema simple de lectura en "formato TAC" y escritura en "formato PC + estructura standard" (PCX, TIFF, etc).

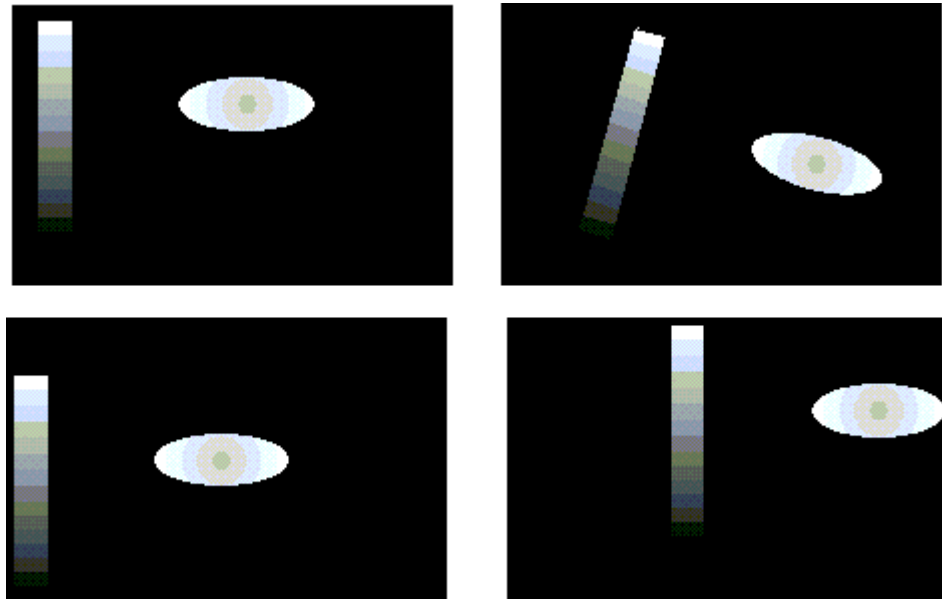
Si se utiliza el sistema de adquisición presentado en (3.1.2), se debería implementar el driver para la placa digitalizadora y simplemente almacenar la información, tal cual viene, en un formato adecuado.

Si se emplea, en cambio, la modalidad de adquisición presentada en (3.1.3), debemos corregir los problemas de registración antes de continuar.

Que queremos decir con "registración"?

En la (Ilustración1-A) se observaba la ubicación espacial de cada uno de los cortes axiales realizados por el tomógrafo, a partir de los cuales se conforman las imágenes bidimensionales.

En la (Ilustración1-B) se observa una imagen bidimensional generada por el tomógrafo que corresponde a un determinado corte de un estudio. Imágenes similares a esta son las que muestra la pantalla del tomógrafo y las que se deberían observar en cada una de nuestras representaciones una vez realizado el proceso de adquisición.

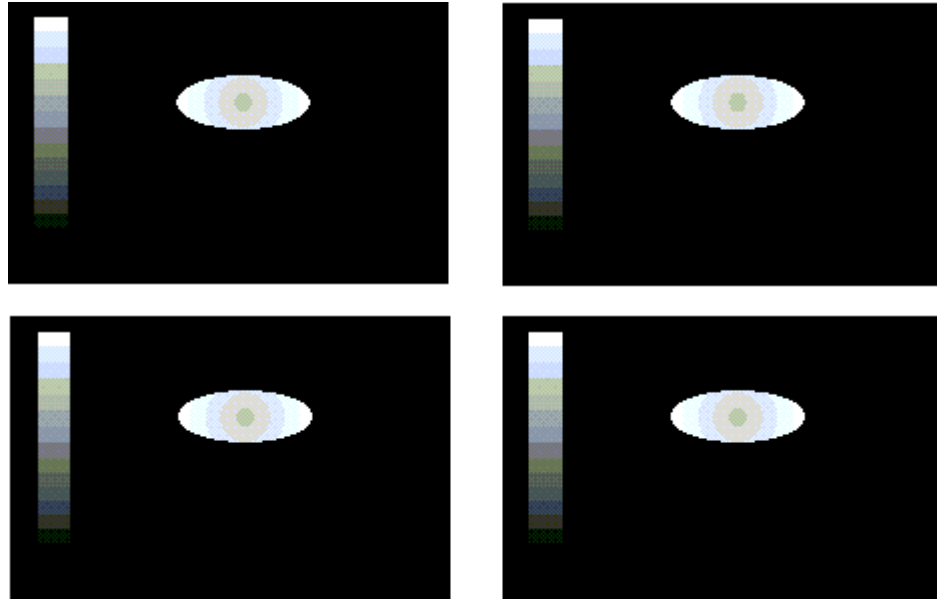


• Ilustración 6A: Resultados posibles luego de scanear cuatro cortes de un objeto cilíndrico de forma elipsoidal

Al scanear las placas, se introducen errores geométricos de traslación y de orientación (rotación) en las imágenes. En la (Ilustración 6-A) se muestran los efectos de estos errores. En esta figura tenemos un objeto elipsoidal hipotético que nos ilustra esquemáticamente la necesidad de referir las imágenes a un sistema coordenado o de punto fijo previo a su procesamiento.

Esta figura da una idea de la importancia de mantener una coherencia espacial entre los distintos estratos o cortes realizados, de modo de poder recrear la imagen tridimensionalmente.

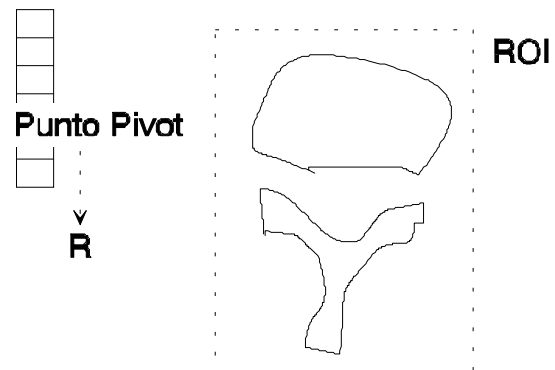
El objetivo primario es, entonces, aplicar una serie de transformaciones a los distintos cortes (slices) de un estudio tomográfico para obtener una colección de imágenes con una suerte de coherencia espacial (Ilustración 6-B).



• Ilustración 6B: Resultados luego de registrar (anclar) las imágenes de 6A

3.2.1 Identificación del Punto en Común (Punto Pivote)

Como primer paso se debe identificar, en cada imagen, un punto "común" (facilmente identificable en todas). Es conveniente seleccionar vértices de ángulos agudos y de alta intensidad, de modo que sea simple distinguirlo en las distintas imágenes (Ilustración 7-A).



• Ilustración 7A: Ubicación del Punto Pivote y la Región de Interés en una imagen hipotética

Al anotar para cada corte la posición (relativa al 0,0 de la imagen -esquina superior izquierda-) del punto pivote, y relacionando todas las medidas y/o cantidades con este punto, en realidad estamos definiendo un sistema global de coordenadas para todos los cortes (origen (x, y) universal).

Este concepto de coordenada de referencia o punto común se ilustra en la (Ilustración 7-B) donde se ha elegido arbitrariamente como punto pivote un borde agudo en la información de referencia de cada placa tomográfica.

Con este procedimiento se "registran" todos los slices del estudio a este punto o sistema global y logramos que una coherencia tridimensional sea posible entre los mismos lo cual permite relacionarlos espacialmente para representar el objeto lógico en 3D.

3.2.2 Determinación de la Región de Interés (ROI)

Es necesario disminuir el volumen de información que se va a procesar, tanto para aislar los posibles objetos de interés, como para aumentar la performance.

Dentro de cada corte se debe filtrar o depurar la imagen para reducir la cantidad de datos a procesar y para eliminar gráficos de control (como barras de intensidades, escalas, nombre, fechas etc.) introducidos por los equipos de TAC.

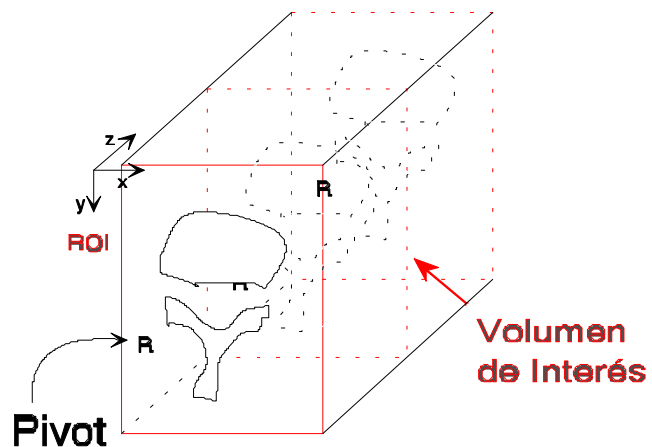
Se selecciona entonces el mínimo rectángulo que contenga toda la información relevante (Ilustración 7-A) y se especifican las coordenadas de este relativas al punto pivote.

Es importante notar que, de haber utilizado cualquiera de los sistemas de adquisición descritos en (3.1.1) o (3.1.2), este proceso se hubiera simplificado mucho, ya que con solo definir un ROI "tipo" sería suficiente (que podría estar parametrizado según el tipo de estudio a encarar, -cabeza, cadera, etc-).

3.2.3 Determinación del Volumen de Interés (VOI)

Al pasar de 2D a 3D es necesario asegurarse de no recortar ninguna parte del objeto que se desea estudiar.

Esto se consigue seleccionando la máxima región de interés de la colección de todas las imágenes bidimensionales, y definiendo así un volumen de interés como la extrusión de este ROI a lo largo de un eje perpendicular a los planos que contienen las imágenes (Ilustración 7-B).



• Ilustración 7B: Ubicación de los ROI y el VOI en el espacio de datos

Se descarta, entonces, todo el material "sobrante" y se define al VOI como la zona donde continuará el estudio. Todo el procesamiento que se ha de realizar a los distintos slices se circunscribe, de ahora en más, al volumen de interés definido.

§ 3.3 Transformaciones de la Imagen

Así como a cada elemento de imagen bidimensional se le denomina pixel (picture element), a cada elemento de espacio tridimensional se lo conoce como voxel. Es decir, cada voxel representa un volumen paralelepípedo de datos, que comúnmente es considerado con aristas de igual longitud, con lo cual se definen voxels cúbicos.

En el caso particular de este trabajo, cada voxel del VOI representa la densidad del tejido pero, según el método de adquisición utilizado es posible que se haya introducido cierto nivel de ruido.

Se busca entonces mejorar las imágenes mediante el filtrado. Existen para este propósito distintos tipos de algoritmos. En las dos secciones siguientes se describe brevemente el funcionamiento de algunas transformaciones y en el párrafo siguiente los filtros y se explican conveniencias en la implementación de los mismos.

3.3.1 Transformada de Fourier

Si $f(x,y)$ es continua e integrable, y $F(u,v)$ es integrable, se tiene que el siguiente par de transformadas de Fourier existen

$$Four\{f(x,y)\} = F(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y) \cdot e^{[-2j\pi(ux+vy)]} dx dy$$

y

$$Four^{-1}\{F(x,y)\} = f(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} F(u,v) \cdot e^{[-2j\pi(ux+vy)]} du dv$$

Donde $j = \sqrt{-1}$ y $Four^{-1}\{F(u,v)\}$ es la transformada inversa o antitransformada de Fourier.

Suponemos que la función $f(x,y)$ es discretizada en una secuencia:

$$\left\{ \begin{array}{l} f(x_0, y_0), f(x_0 + \Delta x, y_0 + \Delta y), f(x_0 + 2\Delta x, y_0 + 2\Delta y) \cdots \\ \cdots f(x_0 + [M-1]\Delta x, y_0 + [M-1]\Delta y) \end{array} \right\}$$

Tomando M intervalos de Δx para x, y N intervalos Δy para y. Para simplificar la notación, de ahora en más usaremos las variables x e y como si fuesen discretas, esto es:

$$f(x,y) = f(x_0 + x\Delta x, y_0 + y\Delta y)$$

Con la notación convenida arriba, tenemos que la transformada discreta de Fourier de un imagen, $f(x,y)$, es otra imagen, $F(u,v)$, cuyos valores son complejos, definidos en una misma región, y se obtienen con la siguiente fórmula:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \cdot e^{-2j\pi \frac{ux+vy}{N}}$$

$$u = 1, 2, 3, \dots, M-1; v = 1, 2, 3, \dots, N-1$$

Se puede demostrar por directa sustitución [GON87], que la transformada discreta de Fourier siempre existe, por lo que no tenemos que preocuparnos en que $f(x, y)$ sea continua e integrable.

En un arreglo cuadrado tenemos que M y N son iguales, lo que permite reducir la fórmula (3.3.1-4) a:

$$F(u, v) = \frac{1}{N^2} \sum_{X=0}^{n-1} \sum_{Y=0}^{n-1} f(x, y) \cdot e^{-2j\pi \frac{ux+vy}{N}}$$

$$u, v = 1, 2, 3, \dots, N-1$$

Puede ser demostrado [GON87] que los incrementos en los intervalos espaciales y de frecuencia están relacionados por:

$$\Delta u = \frac{1}{M \cdot \Delta x}$$

y

$$\Delta v = \frac{1}{M \cdot \Delta y}$$

La transformada de Fourier es particularmente útil en el procesamiento digital de imágenes debido a las propiedades que cumple y a los atributos que posee, tanto ésta como su inversa (Antitransformada de Fourier).

Para simplificar la explicación vamos a desarrollar en cada caso solo la transformada directa de Fourier, dejando al lector la libertad de remitirse a la bibliografía del trabajo [GON87][MAS89] o simplemente hacer la analogía para el caso de la antitransformada. Entre todas las propiedades de la transformada de Fourier, las siguientes son de real importancia en el procesamiento de imágenes:

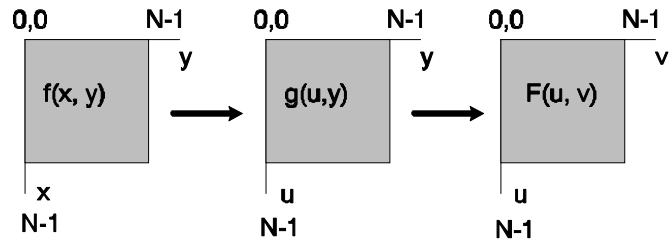
3.3.1.1 Separabilidad

La transformada discreta de Fourier puede ser expresada en una fórmula que permite separarla perfectamente en dos sumatorias :

La utilidad de esta propiedad radica en que $F(u, v)$ puede ser obtenida en dos pasos por sucesivas aplicaciones de la transformada de Fourier unidimensional

$$\begin{aligned}
F(u, v) &= \sum_{y=0}^{N-1} \sum_{x=0}^{N-1} f(x, y) \cdot e^{-2j\pi \frac{ux+v}{N}} \\
&= \sum_{y=0}^{N-1} \left[\sum_{x=0}^{N-1} f(x, y) \cdot e^{-2j\pi \frac{ux}{N}} \right] \cdot e^{-2j\pi \frac{vy}{N}} \\
&= \sum_{y=0}^{N-1} g(u, y) \cdot e^{-2j\pi \frac{vy}{N}}
\end{aligned}$$

En la (Fig. 3.3.1-1) se esquematiza esta propiedad de separabilidad de la transformada de Fourier a través de sucesivas transformadas unidimensionales



• Fig. 3.3.1-1: Cálculo de la Transformada de Fourier bidimensional

3.3.1.2 Translación

La propiedad de translación de la transformada de Fourier está dada por:

$$f(x, y) \cdot e^{-2j\pi \left(\frac{u_0 \cdot x}{M} + \frac{v_0 \cdot y}{N} \right)} \Leftrightarrow F(u - u_0, v - v_0)$$

y

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v) \cdot e^{-2j\pi \left(\frac{u \cdot x_0}{M} + \frac{v \cdot y_0}{N} \right)}$$

Donde la doble flecha es usada para indicar la correspondencia entre la función y su transformada. La ecuación (3.3.1-9A) muestra que multiplicar la función original por el término exponencial indicado y tomar la transformada de este producto, equivale a desplazar el origen del plano de frecuencia al punto (u_0, v_0) ("Modulación")

Eligiendo un punto de desplazamiento (u_0, v_0) apropiado, podemos simplificar considerablemente los cálculos. Si tomamos dicho punto como $(N/2, N/2)$ obtenemos:

$$e^{2j\pi\left(\frac{u_0 \cdot x}{N} + \frac{v_0 \cdot y}{N}\right)} = e^{j\pi(x,y)} \\ = (-1)^{x+y}$$

Por lo tanto, el origen de la transformada de Fourier puede ser desplazado al centro de su correspondiente cuadrado $[N \times N]$ de frecuencia multiplicando $f(x,y)$ por el término $(-1)^{x+y}$.

$$f(x,y)(-1)^{x+y} \Leftrightarrow F\left(u - \frac{N}{2}, v - \frac{N}{2}\right)$$

Es importante notar en la ecuación (3.3.1-9B) que un desplazamiento en $f(x,y)$ no afecta la magnitud de la transformada de Fourier, ya que

$$\left| F(u,v) \cdot e^{-2j\pi\left(\frac{u \cdot x_0}{M} + \frac{v \cdot y_0}{N}\right)} \right| = |F(u,v)|$$

3.3.1.3 Periodicidad y Simetría Conjugada

La transformada discreta de Fourier y su inversa son periódicas de período N . Esto es:

$$F(u,v) = F(u + N, v) = F(u, v + N) = F(u + N, v + N)$$

La validez de esta propiedad puede ser demostrada directamente sustituyendo en la ecuación (3.3.1-11). Solo se necesita un período de la transformación para completar $F(u,v)$ en el dominio espacial.

Si $f(x,y)$ es real, la transformada de Fourier también posee simetría conjugada, ya que:

$$F(u,v) = F^*(-u,-v)$$

o más interesante aún

$$|F(u,v)| = |F^*(-u,-v)|$$

3.3.1.4 Rotación

Si introducimos coordenadas polares: $u = w \cdot \cos(\phi)$, $v = w \cdot \sin(\phi)$, $x = r \cdot \cos(\theta)$ y $y = r \cdot \sin(\theta)$; $f(x,y)$ y $F(u,v)$ ahora son $f(r,\theta)$ y $F(w,\phi)$, respectivamente. Se puede demostrar por directa sustitución, ya sea en la transformada de Fourier continua o discreta, que

$$f(r, \theta + \alpha) \Leftrightarrow F(w, \phi + \alpha)$$

En otras palabras, si $f(x,y)$ es rotada un ángulo α ; $F(u,v)$ es rotada el mismo ángulo.

3.3.1.5 Distribución y Escalamiento (Linearidad)

Es consecuencia directa de la definición de transformada, continua o discreta, que

$$f_1(x, y) + f_2(x, y) \Leftrightarrow \text{Four}\{f_1(x, y)\} + \text{Four}\{f_2(x, y)\}$$

y, en general, que

$$\text{Four}\{f_1(x, y) \cdot f_2(x, y)\} \neq \text{Four}\{f_1(x, y)\} \cdot \text{Four}\{f_2(x, y)\}$$

En otras palabras, la transformada de Fourier (y su inversa) son distributivas respecto a la suma pero no a la multiplicación.

Se puede demostrar también, [GON87] que para dos escalares a y b ,

$$a \cdot f(x, y) \Leftrightarrow a \cdot F(u, v)$$

y

$$f(a \cdot x, b \cdot y) \Leftrightarrow \frac{1}{|a \cdot b|} F\left(\frac{u}{a}, \frac{v}{b}\right)$$

3.3.1.6 Laplaciano

El Laplaciano de una función de dos variables, $f(x, y)$, está definido como

$$\nabla^2 f(x, y) = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}$$

Es consecuencia de la definición de transformada bidimensional de Fourier que

$$\text{Four}\{\nabla^2 f(x, y)\} \Leftrightarrow -(2\pi)^2 \cdot (u^2 + v^2) \cdot F(u, v)$$

El operador Laplaciano es útil para detectar bordes en una imagen, como se verá mas adelante.

3.3.1.7 Convolución y Correlación

Consideraremos dos relaciones entre transformadas de Fourier que constituyen un nexo básico entre los dominios espaciales y de frecuencia. Estas relaciones, llamadas convolución y correlación, son de fundamental importancia para comprender las técnicas de procesamiento de imágenes basadas en la transformada de Fourier.

El producto de convolución de dos funciones discretas, $f(x, y)$ y $g(x, y)$, denotado $f(x, y) \bullet g(x, y)$, se define como

$$f(x, y) \bullet g(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) \cdot g(x-m, y-n)$$

$$x = 1, 2, 3, \dots, M-1; y = 1, 2, 3, \dots, N-1$$

La importancia de la convolución en el dominio de frecuencia radica en el hecho de que; si $f(x, y)$ tiene como transformada de Fourier a $F(u, v)$ y $g(x, y)$ a $G(u, v)$, entonces $f(x, y) \bullet g(x, y)$ tiene como transformada $F(u, v)G(u, v)$. Formalmente,

$$f(x, y) \bullet g(x, y) \Leftrightarrow F(u, v) \cdot G(u, v)$$

Este principio es conocido como *Teorema de Convolución*.

Utilizando este es posible calcular la convolución de dos funciones multiplicando solo sus transformadas que, eventualmente, podrían ser obtenidas con el algoritmo de la Transformada Rápida de Fourier(FFT)[GON87].

La correlación de dos funciones $f(x, y)$ y $g(x, y)$, expresadas en forma discreta, se denota como $f(x, y) \circ g(x, y)$ y está definida por la fórmula

$$f(x, y) \circ g(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n) \cdot g(x-m, y-n)$$

$$x = 1, 2, 3, \dots, M-1; y = 1, 2, 3, \dots, N-1$$

Donde f^* significa conjugada de f .

Si $f(x, y)$ y $g(x, y)$ son iguales, la ecuación (3.3.1-25) se la conoce como autocorrelación.

Es posible demostrar lo que sostiene el Teorema de Correlación tanto para el caso continuo como para el discreto

$$f(x, y) \circ g(x, y) \Leftrightarrow F(u, v) \cdot G(u, v)$$

y

$$f(x, y) \cdot g(x, y) \Leftrightarrow F(u, v) \circ G(u, v)$$

Una de las principales aplicaciones de la correlación en el procesamiento digital de imágenes es en búsqueda del patrón apropiado de encaje (prototype matching), donde el problema es encontrar la coherencia espacial entre una imagen desconocida y un conjunto de imágenes con origen conocido. Una aproximación es calcular la correlación entre la imagen desconocida y cada una de las imágenes prototipos. El prototipo mas acorde con la imagen desconocida es aquel con el que tiene una función de correlación mayor.

3.3.2 Transformada Discreta del Coseno

La transformada discreta del coseno está definida por el par

$$C(0,0) = \frac{2}{M \cdot N} \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} f(j,k)$$

$$C(m,n) = \frac{4}{M \cdot N} \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} f(j,k) \cdot \cos\left[(2j+1)\frac{\pi m}{2M}\right] \cdot \cos\left[(2k+1)\frac{\pi n}{2N}\right]$$

$$m = 1, 2, 3, \dots, M-1; n = 1, 2, 3, \dots, N-1$$

Como tiene, al igual que la transformada de Fourier, la propiedad de ser separable, puede ser implementada por sucesivas transformaciones unidimensionales.

§ 3.4 Mejoramiento de la Imagen

El principal objetivo de las técnicas de mejoramiento de la imagen (image enhancement) es procesar una imagen dada de modo tal que el resultado sea más apropiado, para una aplicación específica, que la imagen original. La palabra "específica" es importante ya que caracteriza de antemano a las técnicas que se verán a continuación, como metodologías orientadas al problema. Por lo tanto, un método que es bastante útil para mejorar las imágenes generadas por un TAC, puede no ser necesariamente la aproximación más óptima para mejorar las imágenes enviadas por un satélite.

Las técnicas para mejoramiento de la imagen puede ser clasificada en dos grandes categorías:

- METODOS EN EL DOMINIO DE FRECUENCIA: Basados en la modificación de la transformada de Fourier de la imagen.
- METODOS EN EL DOMINIO ESPACIAL: Se refieren al plano de la imagen en si mismo, y las aproximaciones en esta categoría están basadas directamente en la manipulación de los pixels de la imagen.

3.4.1 Métodos en el Dominio de Frecuencia

El fundamento de las técnicas basadas en el dominio de frecuencia es el teorema de convolución.

Sea $g(x,y)$ una imagen formada por la convolución de una imagen $f(x,y)$ y un operador invariante por posición $h(x,y)$ (cuyo resultado depende solo del valor de $f(x,y)$ en un punto dado de la imagen y nó de la posición de ese punto). Esto es,

$$g(x,y) = h(x,y) \bullet f(x,y)$$

Entonces, por el teorema de convolución, se tiene que la siguiente relación en el dominio de frecuencia se mantiene:

$$G(u,v) = H(u,v) \cdot F(u,v)$$

Donde G , H , F son las transformadas de Fourier de g , h , f , respectivamente. La transformada $H(u,v)$ es llamada a veces "función de transferencia".

Numerosos problemas de mejoramiento de imagen pueden ser expresados con la fórmula (3.4.1-2). En una típica aplicación de mejoramiento de imagen, $f(x,y)$ es dada y el

objetivo, después del cálculo de $F(u,v)$ es seleccionar un $H(u,v)$ tal que la imagen deseada, dada por,

$$g(x,y) = \text{Four}^{-1}\{H(u,v) \cdot F(u,v)\}$$

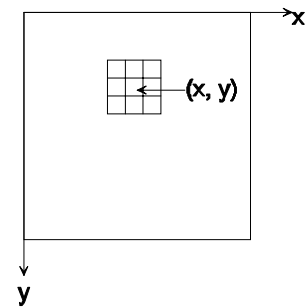
resalte o mejore alguna característica de $f(x,y)$. Por ejemplo, los bordes pueden ser acentuados usando una función $H(u,v)$ que aumente la componente de $F(u,v)$ de alta frecuencia [GON87].

3.4.2 Métodos en el Dominio Espacial

El término dominio espacial se refiere al conjunto de pixels que componen una imagen y los métodos en el dominio espacial son procedimientos que operan directamente en esos pixels. Las funciones de procesamiento de imágenes en el dominio espacial pueden ser expresadas como:

$$g(x,y) = T(f(x,y))$$

donde $f(x,y)$ es la imagen de entrada, $g(x,y)$ es la imagen procesada y T es un operador en f , siendo el soporte de la función T o f un entorno de (x,y) . La mejor aproximación para definir el significado de vecinos de un pixel (x,y) es usar una imagen cuadrada o rectangular (vecindad) centrada en (x,y) , como se muestra en la (Fig.3.4.2-1). El centro de la subimagen es movido de pixel a pixel comenzando, por ejemplo, en la esquina superior izquierda, y aplicando el operador a cada posición (x,y) para obtener el valor de $g(x,y)$ correspondiente a esa posición. Aunque otras formas de vecindad, tal como un círculo, son a veces usadas, los arreglos cuadrados son los que predominan debido a la facilidad de implementación.



• Fig. 3.4.2-1: Vecindad-8 de un punto (x, y)

La forma más simple de T es cuando la vecindad es de 1×1 . En esos casos g depende solo del valor de f en (x,y) , T consiste en una transformación en la escala de gris y se la denomina función de mapeamiento.

Al considerar una vecindad mayor, se pueden definir una mayor variedad de transformaciones o filtros que cumplan la función de mejorar la calidad de la imagen para un propósito determinado. Una de las aproximaciones más comunes a los filtros digitales son las máscaras (también llamadas ventanas). Una máscara es un pequeño arreglo (p.e. 3×3) cuyos coeficientes son elegidos para detectar alguna propiedad en la imagen. Así, dándole distintos valores a los coeficientes se puede obtener filtros para detectar bordes

en distintas direcciones [MAS89], el filtro de la media, que suaviza la imagen, el filtro de la mediana que resalta los contornos, etc.

La elección de un filtro u otro depende, como ya se dijo anteriormente, de la propiedad de la imagen que se desea obtener. Posiblemente no haya un único filtro apropiado para una determinada propiedad; la imagen deberá ser sometida a sucesivos filtros para su mejoramiento general. De esto se deduce que muchas veces encontrar el filtro óptimo se convierte en una tarea experimental de combinación de varias máscaras.

A continuación se desarrollarán filtros que, por su simplicidad en la implementación, su bajo costo computacional y su aceptable performance, son frecuentemente usados en el proceso de mejoramiento de imagen. Luego se explicarán los fundamentos que motivaron la elección del filtro en el presente trabajo.

3.4.2.1 Filtro de la Media

El filtro de la media consiste en el promedio entre cada punto $P(x,y)$ y los puntos en su vecindad-8. La formulación de este filtro es la siguiente

$$f(x,y) = \frac{1}{M} \sum_{m,n \in S} f(m,n)$$

$$x, y = 1, 2, 3, \dots, M - 1$$

Donde S es la colección de puntos que conforman la vecindad-8.

Este filtro tiende a suavizar las discontinuidades, algo que no es propicio para los fines del trabajo ya que luego nos deberemos basar en éstas para determinar los bordes del objeto. Si bien a la vista es mas atractivo (la imagen parece mas "continua"), es preferible elegir un filtro que no "borre" los bordes.

Un resultado similar se obtiene si se utiliza una máscara con todos los coeficientes del arreglo igual a $1/9$:

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

• Filtro de la Media: máscara de 3x3

3.4.2.2 Filtro de la Mediana

La principal dificultad del método descrito en la sección anterior es que suaviza bordes y otros detalles propios de la imagen. Aunque ese problema puede ser de algún modo salvado usando una intensidad mínima como umbral. Es decir, todo pixel de la imagen con intensidad menor que el umbral no es considerado parte del objeto de interés. La elección de un valor de umbral apropiado suele convertirse en una tarea experimental relativamente ardua. Una aproximación alternativa es usar el filtro de la mediana por el cual se reemplaza el nivel de gris de cada pixel por la mediana de los tonos de grises del pixel y sus vecinos, en vez del promedio.

Este método es particularmente útil cuando la característica de agudeza y definición precisa de bordes desea ser conservada.

A continuación se explica el funcionamiento de este filtro:

Para un pixel cualquiera $P(x,y)$ hacer una lista de los pixels en su vecindad-8 $\{(x-1,y-1), (x,y-1), (x+1,y-1), (x-1,y), (x+1,y), (x-1,y+1), (x,y+1), (x+1,y+1)\}$ y el pixel en cuestión, ordenar y seleccionar como valor de P al quinto valor de la lista.

Para los objetivos propuestos en este trabajo es muy conveniente emplear el filtro de la mediana ya que elimina los picos de intensidad (probablemente producto del ruido) y a la vez conserva la necesaria definición de bordes a los efectos de aislar un objeto determinado.

CAPITULO IV

REPRESENTACION

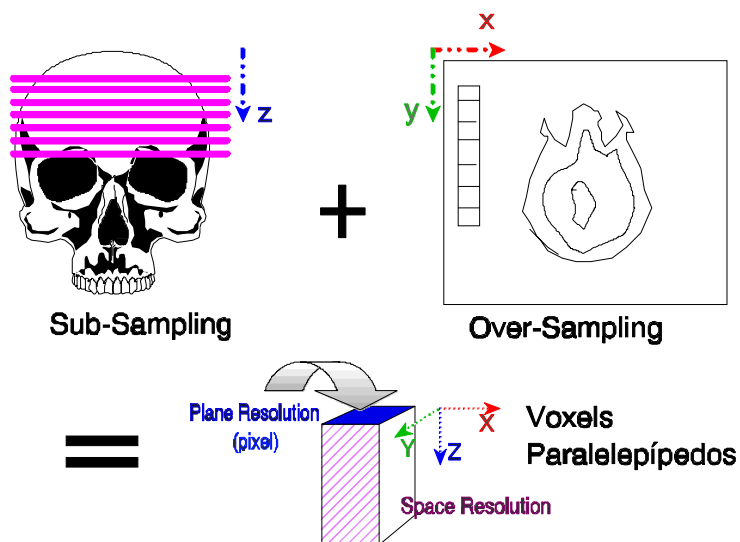
Por representación entendemos las distintas formas de modelar un objeto. Esto es, las distintas estructuras de datos que nos permiten guardar las características de un objeto tridimensional en la memoria de la computadora.

§ 4.1 Obtención de Voxels Cúbicos

Las mediciones realizadas por los equipos TAC son "puntuales". O sea, obtienen el valor de la densidad del tejido en el punto de aplicación. Sin embargo, debido a condicionamientos técnicos obtenemos una mayor densidad de mediciones sobre un "plano" (o corte) determinado que cantidad de cortes sobre el objeto de estudio.

Es muy conveniente trabajar con voxels cúbicos. Esto permite una mejor triangulación de los datos, mayor suavidad en las superficies, como así también una selección más inteligente de los voxels de interés.

El conjunto de datos con los que se trabajará tiene una resolución en el plano X-Y de la imagen mucho mayor que la resolución en Z, que esta en función de la cantidad de cortes del estudio (o número de slices)..



• Ilustración 4-1: Efectos de Subsampling en Z y Oversampling en X,Y

Cada valor dentro del conjunto de datos representa, en el caso de TAC, la densidad de tejido en ese punto. Observando la (Figura 4-1) se aprecia que el resultado del Sub-Sampling supone la misma densidad a lo largo de todo el espacio entre un corte y el siguiente, del cual se carece de información.

Teniendo en cuenta que la resolución en el plano X-Y varía normalmente entre los 0,4 y 1,5 milímetros y la separación entre cortes (resolución en Z), va de 2mm, en el mejor de los casos, a 10 mm, se tiene la siguiente relación:

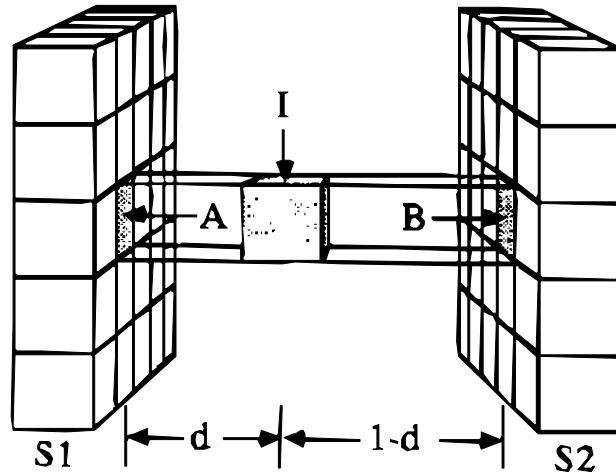
	Max(xy)-Max(z)	Max(xy)-Min(z)	Min(xy)-Max(z)	Min(xy)-Min(z)
Res-XY (mm)	0.4	0.4	1.5	1.5
Res-Z (mm)	2	10	2	10
Voxel Cúbico (mm ³)	0.06	0.06	3.38	3.38
Voxel Muestra (mm ³)	0.32	1.6	4.5	22.5
Error (mm ³)	0.26	1.54	1.13	19.13
% Error	400	2400	33.33	566.67
Cortes Artificiales	5	25	1.3	6.6

• Combinaciones de Resoluciones Máximas y Mínimas Posibles en un estudio de Tomografía Computada

Es decir, el error del volumen estimado con el volumen cúbico deseado va, desde 33.33% en el mejor de los casos (cuando la resolución en pantalla es mínima y la cantidad de cortes máxima), a 2400 % en el peor de los casos

4.1.1 Interpolación Lineal

La cantidad de cortes a generar mediante interpolación se obtiene dividiendo el espacio entre cortes por la resolución en XY, el plano transversal.



• Fig. 4-2: Disposición espacial de dos cortes muestrados y el voxel a ser estimado

Se tiene entonces la situación mostrada en la (Figura 4-2). Es necesario determinar la intensidad del voxel intermedio.

El método clásico es interpolar linealmente las intensidades de los voxel extremos [RAY90]. Esta interpolación es función de la posición y es adecuada para lograr aproximaciones simples. Más adelante se detallan los problemas relacionados con esta aproximación.

Este algoritmo de interpolación con algunos valores de ejemplo, resultaría en algo similar a lo mostrado en la (Figura 4-3).

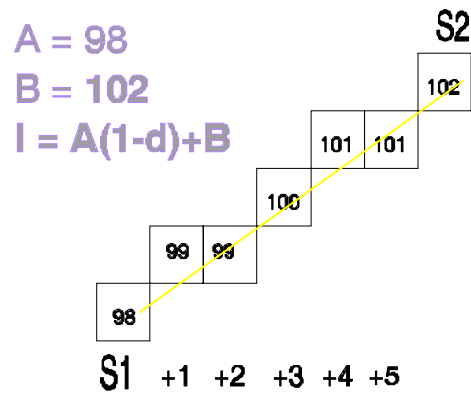
La función de interpolación viene dada por:

$$I = A \cdot (1 - d) + B \cdot d$$

Donde A y B son las intensidades medidas de los voxels en los slices extremos y d es la distancia (normalizada) entre el voxel con intensidad A y el voxel a interpolar.

Este método para obtener más cortes, si bien es adecuado para una primera aproximación tiene la desventaja de no mejorar significativamente la suavidad de las superficies una vez reconocidos los objetos.

Algoritmo de Interpolación



• Fig. 4-3: Algoritmo de Interpolación Lineal con valores de ejemplo para voxels extremos A y B

En los bordes de los objetos, se presentan abruptos cambios en la intensidad de los voxels de distintos cortes. Este abrupto cambio de intensidad, acoplado con el método de reconocimiento de formas mas utilizado, el del umbral, tiende a producir superficies con bordes abruptos o discontinuidades.

A modo de ejemplo, supóngase tener dos cortes muestreados con los siguientes valores de intensidad:

Slice n	199	189	145	83	33	11	11	14	14	12
Slice n+1	164	166	164	164	174	188	183	139	69	14

• Fig. 4-3: Valores ejemplos de dos slices consecutivos con un abrupto cambio en el borde (debido a subsampling)

El borde mostrado divide las regiones de objeto y no-objeto, habiendose utilizado 136 como valor de umbral para la desición.

Se puede observar lo abrupto del cambio de posición del borde (debido a la cantidad de separación de los cortes en relación con la velocidad de cambio de la superficie en esa zona). Mientras que en el corte n, la posición del borde estaría en el tercer voxel, en el corte siguiente la posición cambia al voxel octavo, provocando el clásico escalonamiento.

Sin embargo, interpolando linealmente los cortes mostrados y utilizando 136 nuevamente como valor umbral, se tendría lo siguiente:

Slice n	199	189	145	83	33	11	11	14	14	12
Slice interpolado 1	192	184	149	99	61	46	45	39	25	12
Slice interpolado 2	185	180	153	115	89	82	79	64	36	13
Slice interpolado 3	178	175	156	132	118	117	114	89	47	13
Slice interpolado 4	171	171	160	148	146	153	148	114	58	14
Slice n+1	164	166	164	164	174	188	183	139	69	14

• Fig. 4-4: Resultados de interpolar entre los slices mostrados en (4-3) y aplicar un filtro de umbral con valor de desición 136.

Se observa que la interpolación lineal directa de intensidades no ha mejorado en mucho la discontinuidad en el borde. Esto es el resultado de utilizar un método de reconocimiento simple como el de umbral, pero éste es el más usado en aplicaciones de medical imaging, que son útiles para éste trabajo.

Recientemente se han presentado [HER92] trabajos para obtener los cortes interpolados sin sufrir este problema. Estos métodos, basados en el estudio de las discontinuidades luego de reconocer los bordes en cada corte, son muy recientes y, por lo tanto, no fueron evaluados por nosotros como para presentar un juicio de valor de los mismos.

4.1.2 Espacio de Almacenamiento de Datos

Un aspecto importante que hay que tener en cuenta en el proceso de obtención del volumen cúbico de datos es el del almacenamiento. La generación de slices adicionales a los del estudio original significa un aumento muy importante en la cantidad de datos del modelo. Teniendo en cuenta las resoluciones en los tres planos antes mencionadas, se pueden crear entre 2 y 25 cortes adicionales entre cada par de cortes muestreados. Si cada corte tiene aproximadamente 512 x 512 pixels (512 x 512 bytes, suponiendo 1 byte por pixel) de resolución en X,Y, y un estudio simple se compone de 15 a 20 placas, se tendrían los siguientes requerimientos de almacenamiento:

$$A = \text{ResX} * \text{ResY} (N + (N - 1) * C_{\text{Interp}})$$

$$N = \text{Número de cortes muestreado}$$

$$\text{ResX} = \text{Resolución (cantidad de pixels) en X}$$

$$\text{ResY} = \text{Resolución (cantidad de pixels) en Y}$$

$$C_{\text{Interp}} = \text{Cantidad de Cortes artificiales a generar entre cada par de cortes}$$

Los requerimientos mínimos y máximos para almacenar 15 y 20 placas entre 2 y 25 cortes adicionales serían los siguientes:

$$(\text{mejor caso}) A = 512 * 512 * (15 + (15 - 1) * 2) = 11272192 \text{ Bytes (11MB)}$$

$$(\text{peor caso}) A = 512 * 512 * (20 + (20 - 1) * 25) = 129761280 \text{ Bytes (124MB)}$$

Si bien estos datos pueden ser compactados de alguna manera, en algún momento deben ser procesados. Es por esto que para trabajar con Volume Rendering - que se basa en el procesamiento de cada elemento de volumen-, en oposición a Surface Rendering -que se basa en el procesamiento de entidades matemáticas que representan al volumen-, se necesitan workstations especiales con gran velocidad de proceso. (Los conceptos de Volumen y Surface Rendering son explicados con mayor detalle en el párrafo § 4.3.)

Es importante hacer notar que en todos los cálculos arriba expuestos, se asumió que un voxel equivale a un byte de almacenamiento (8 bits). Si bien esta es una suposición razonable (especialmente en nuestro caso en que la adquisición de datos fue realizada mediante equipos "entorno PC" -que generalmente trabajan con resoluciones de 1 byte por pixel-) no es del todo exacta, ya que la mayoría de los equipos de TAC trabajan con una resolución de 12 bits (1.5 byte) por voxel (pixel).

§ 4.2 Identificación del órgano de interés

La utilidad de las distintas técnicas de medical imaging radica en la posibilidad de poder hacer una visualización no-invasiva del órgano de interés. Hasta el momento, no tenemos en nuestro volumen de datos más que una serie de mediciones de la densidad del tejido en el punto de medición.

Si bien, como se verá mas adelante, estos datos permitirían hacer cualquier tipo de cortes (axiales, sagitales, coronales o en cualquier orientación), extendiendo notablemente la posibilidad de los equipos de TAC estándares, el objetivo es obtener, de todo el volumen de datos, el órgano que estamos interesados en estudiar.

La tarea de reconocer un objeto dentro del volumen podría resumirse en:

Dado un conjunto de voxels (C),

- 1. aplicar una transformación (S) para*
- 2. obtener el subconjunto O , tal que*
- 3. $O \subseteq C$, y*

O sea el conjunto de voxels que componen el objeto de interés

El conjunto C de voxels será el volumen sobre el cual se trabajará. En C , cada voxel tiene el valor de intensidad medido por el equipo TAC (o generado por interpolación).

El conjunto O de voxels es el resultado deseado. Este conjunto puede ser del mismo tamaño o menor que C . Puede estar implementado (como se hace a menudo) como un escena binaria (binary scene), donde O es del mismo tamaño que C , con la salvedad que todos los voxels pertenecientes al objeto se marcan como "OBJETO" y el resto como "NO OBJETO".

La creación de S , la función de segmentación, no es una tarea simple. Por el contrario, es un área en constante investigación y progreso sobre la cual todavía no se ha podido dar una respuesta definitiva (o aún mas, satisfactoria) [HER83].

En el área de medical imaging, los intentos se pueden resumir en:

- Atlas
- Segmentación Interactiva
- Threshold (umbral)

4.2.1 Atlas

Este método emplea una "biblioteca" de imágenes "verdaderas". O sea, se generan, con validación de los especialistas en el área, un serie de imágenes de TAC de distintas partes del cuerpo a distintos niveles, señalándose en cada una los distintos objetos que la componen (por ejemplo, hueso, piel, etc). Luego de generada esta biblioteca, se parametrizan ciertos aspectos de las imágenes (como ser, tamaño longitudinal del cráneo) para luego poder hacer la comparación con las tomografías de los distintos pacientes.

Esta metodología es el intento más completo y complejo de los tres arriba mencionados. Sin embargo, adolece de ciertas dificultades importantes:

- Hay que decidir sobre qué cuerpo se basa la "biblioteca". En principio, sería de interés utilizar un cuerpo "normal". Decidir ¿qué es normal? no es trivial.
- Si bien las distintas personas presentan similitudes notables en ciertos aspectos, pueden ser muy distintos en otros. Es así que la selección de los parámetros para encuadrar las placas de los pacientes con la "biblioteca" no es simple y puede llevar a errores de segmentación graves.
- Finalmente la tarea en sí misma de lograr el encaje (matching) entre una placa cualquiera con la de la biblioteca es muy complicada y lleva demasiado tiempo, debiéndose utilizar transformaciones elásticas no lineales [MAG91].

Algunos trabajos han intentado introducir marcadores en el campo de visión para que sean registrados por el equipo TAC, y luego puedan ser utilizados como parámetros "índice" de la biblioteca. Se han logrado resultados importantes restringiendo el espectro de usos posibles (por ejemplo, solamente estudios de cadera).

4.2.2 Segmentación Interactiva

La segmentación interactiva implica la ayuda de un especialista en las imágenes que están siendo estudiadas. Se van presentando las imágenes de los distintos cortes en forma sucesiva y el especialista (radiólogo, médico, etc) marca con algún dispositivo de entrada las zonas pertenecientes al objeto.

Este método es el que tiene el menor porcentaje de error, ya que utiliza el conocimiento del experto para segmentar las imágenes. A menudo, se usa como base de referencia sobre la cual comparar todas las demás imágenes.

La mayor complicación de este método es la necesidad de disponer de un especialista en diagnóstico por imágenes para cada estudio (y por cada imagen generada por el equipo de TAC). Debido a esto, la segmentación interactiva se utiliza normalmente en la generación de volúmenes de datos "maestros" (por ejemplo, para formar la "biblioteca" mencionada en 4.2.1).

En el caso particular del presente trabajo, se utilizó un cierto nivel de interactividad para solucionar los problemas introducidos por la adquisición de las imágenes mediante scanner. Sin embargo, nuestra "segmentación" se limitó a separar las partes no corpóreas de la imagen (barras de control, letras, etc), dejando la segmentación dentro del cuerpo para el método que se verá a continuación.

4.2.3 Threshold (Filtro de Umbral)

La segmentación por threshold (umbral) es la más simple y la más utilizada en la identificación de hueso, que es el órgano de interés abordado en este trabajo. Además, tiene resultados aceptables y requerimientos computacionales modestos.

La idea detrás de este algoritmo es clasificar cada elemento de volumen según sus propiedades intrínsecas, medidas por el tomógrafo, como OBJETO o NO-OBJETO (parte del órgano o nó). Generalmente se determinan, de forma experimental, valores de umbral adecuados para cada tipo de órgano y/o para cada estudio distinto. Vale la pena hacer notar que, si bien la determinación exacta de los umbrales requiere cierto nivel de interactividad, puede ser realizado muy rápidamente en la misma consola por el operador del equipo TAC.

Una vez determinados los umbrales el algoritmo es simple, y consiste en [FU88]:

Para todos los voxels de C, $C(x,y,z)$

Si $UMBRAL_MINIMO \leq C(x,y,z) \leq UMBRAL_MAXIMO$

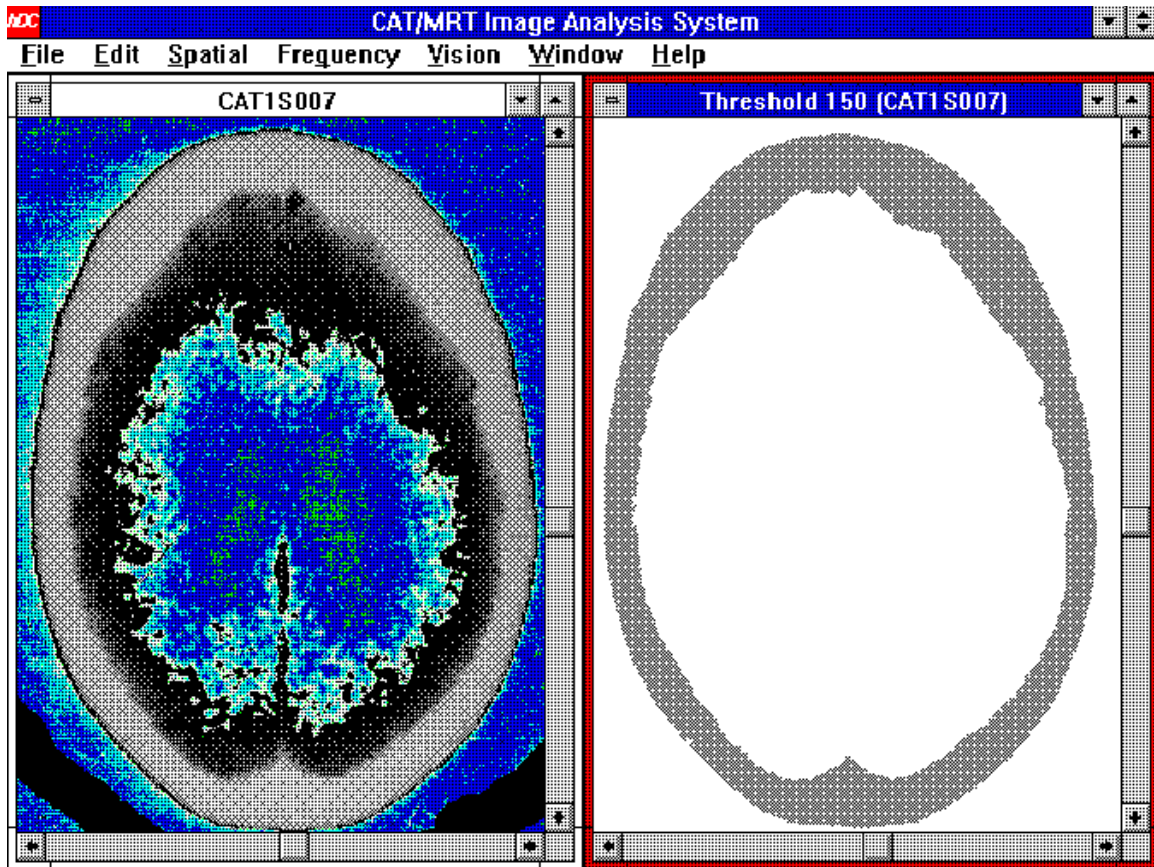
entonces $C(x,y,z) \rightarrow OBJETO$

sinó $C(x,y,z) \rightarrow NO_OBJETO$

Este algoritmo, aplicado a una imagen de TAC de la parte superior del craneo, nos da como resultado algo similar a lo mostrado en la (Figura 4-5).

Todos los métodos arriba mencionados generan como resultado una escena binaria, definiéndose esta como el conjunto de voxels del volumen de datos, previamente procesados de manera tal que cada voxel tenga uno de dos valores posibles (0 o 1), en función de su pertenencia al objeto de interés.

Una extensión natural de esta representación es la compactación de los datos para disminuir el espacio requerido de almacenamiento.



• Fig. 4-5: Resultado de aplicar el filtro de umbral a una imagen obtenida utilizando el método de adquisición descrito en (3.1.3)

Suponiendo siempre una resolución por voxel de 8 bits (1 byte), los requerimientos de espacio se reducirían, entonces, a 1/8 del total, empaquetando 8 voxels por byte. Esta disminución del espacio necesario dejaría el rango de espacio necesario (según se vió en 4.1.2) entre 1.4 y 15.5 MB, lo cual es bastante más aceptable. Es importante entender que, a diferencia de otros algoritmos de compactación, esta reducción permite identificar biunívocamente cada voxel del conjunto a partir de su posición en el espacio (o viceversa, determinar la posición en el espacio de un voxel dado a partir de las resoluciones en los tres planos y su posición dentro del archivo). Los métodos de compactación de datos estándares (LZW, RLL, etc) obtendrían porcentajes de ahorro de espacio mucho mayores, pero tienen el gran inconveniente que para acceder a cualquier voxel deben descompactar todo el archivo (o al menos una parte del mismo).

§ 4.3 Estructuras de Datos

En la escena binaria tenemos toda la información volumétrica que necesitamos para estudiar el objeto de interés.

Dentro del area de Visualización Científica (Scientific Visualization) existen dos grandes ramas de representación de objetos tridimensionales:

- Volume rendering (Basados en volúmen)
- Surface rendering (Basados en superficies).

4.3.1 Volume Rendering

La visualización por volume rendering se basa en la escena binaria (o inclusive, en los datos originales sin hacer la segmentación, o haciéndola en el momento del display) [FRI85] y es una idea conceptualmente muy atrayente. No se intenta describir al objeto de ninguna manera, simplemente se lo divide en elementos cúbicos muy pequeños (como se vió en 4.1), pudiendose direccionar (esto es, clasificar, esconder, modificar) cada uno de estos. Para lograr la graficación tridimensional de algún objeto mediante este método simplemente se grafican, en el orden adecuado, todos los voxels que lo componen.

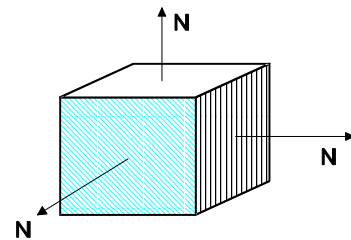
El volume rendering es una disciplina con mucho futuro, ya que las capacidades computacionales aumentan a un ritmo impresionante. Como herramienta para la medición, se la considera más exacta que los métodos mas "inteligentes" (que se verán a continuación), ya que describe todos y cada uno de los elementos que componen a un objeto [MIH91] [NIE91] [FOL90].

4.3.2 Surface Rendering

Los métodos de Surface Rendering describen el objeto mediante aproximaciones matemáticas a las superficies que lo componen. En estos métodos basados en superficies no se distinguen los cambios de intensidad (o densidad) dentro del mismo objeto (ya que se basan en la escena binaria, y esta no preserva el rango de intensidades provista por el equipo TAC),

Existen distintas maneras de representación de superficies en el espacio [BAR82] [FOL90], pero todas se basan en la "figura" (o silueta) del objeto. Las ventajas de estos métodos radican principalmente en la disminución del volumen de datos necesario para describir el objeto, la manejabilidad y maleabilidad de estas estructuras matemáticas y la posibilidad de implementación en equipos de menor porte que los necesarios para los métodos basados en volume rendering.

Existen también estructuras de datos "híbridas", con componentes de ambas metodologías. Una muy utilizada [FRI885] [RAY90] [HER88] es la de generar el volumen completo de voxels y mantenerlos en "memoria", pero para el display se aproxima el voxel por superficies planas cuadradas (Figura 4-6). Este método genera gran cantidad de superficies (1 por voxel, como mínimo) pero es simple de implementar, ya que todas las superficies son simples de obtener y calcular.



• Fig. 4-6: Voxel cúbico con las superficies planas y sus normales

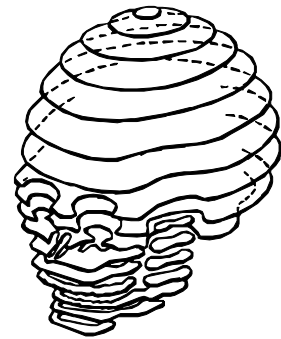
Debido al tipo de equipamiento utilizado, nuestra investigación se centró en los métodos de surface rendering.

Dentro de esta línea, definimos dos tipos principales de estructuras de datos:

- Contornos dirigidos nivelados (Leveled Directed Contours)
- Parches triangulares (Triangular Patches).

4.3.2.1 Contornos Dirigidos Nivelados

Esta estructura de datos representa los bordes del objeto en los distintos cortes (Figura 4-7). Teniendo esta información para cada slice podemos luego utilizarla como guía para derivar otras representaciones mas "complejas" o "completas" del objeto.

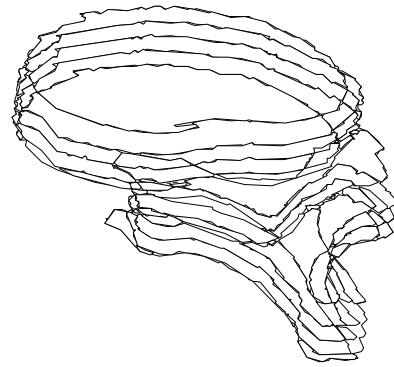


• Fig. 4-7: Esquema de los Contornos Dirigidos Nivelados de un cráneo humano (15 cortes/niveles)

Una de las aplicaciones mas interesantes de esta estructura de datos es usarla como guía para aproximar la superficie del objeto mediante superficies bicúbicas spline [FOL91] [BAR82]. Este tipo de superficies matemáticas necesitan un conjunto de puntos entre los cuales interpolar. Generan una superficie C^1 continua y, por lo tanto, vistas muy atractivas del objeto. Además, ya que las superficies interpolan suavemente por si mismas entre los distintos cortes, se podría obviar el paso de interpolación de intensidades antes descrito.

Podría argumentarse que la interpolación es necesaria, igualmente, para tener mayor "densidad" de puntos guía sobre los cuales basar las curvas, pero basados en trabajos anteriores [BAR82], una mayor densidad en zonas de baja curvatura no mejora la imagen y sí aumenta la carga computacional.

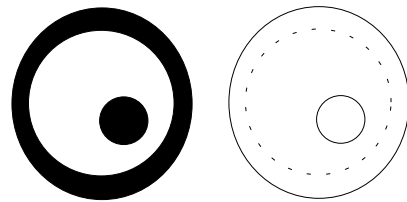
Existen muchos paquetes comerciales de software que hacen el mapping de superficies suaves sobre un conjunto cerrado de puntos. En la (Figura 4-7) podemos observar la disposición espacial de los contornos nivelados. En la (Figura 4-8) vemos la representación producida por nuestro sistema de una vertebra por medio de contornos nivelados.



• Fig. 4-8: Representación por medio de Contornos Dirigidos Nivelados de una vertebra humana (5 cortes)

La construcción de la representación por contornos dirigidos nivelados tiene los siguientes pasos:

1. Se genera la escena binaria. Se tienen N cortes (generalmente solo los muestreados por el TAC, sin recurrir a la interpolación, ya que no nos interesa tener una muy alta densidad en Z) del objeto. Cada corte corresponde a un nivel, y puede haber mas de un contorno por nivel (posiblemente representando los limites exteriores e interiores del objeto) (Figura 4-9)



• Fig. 4-9: (izq) Corte de un objeto hipotético formado por un cilindro hueco con un cilindro mas pequeño macizo adentro. (der) Contornos cerrados identificados, contornos externos con línea sólida e internos con línea de puntos.

1. Dentro de cada corte, se marcan las fronteras (transiciones de estado). Se puede utilizar la máscara de Laplace (Figura 4-10) para realizar esta función, pero es mucho mas práctico utilizar una simple máquina de estados y así mejorar la performance en un 70% aproximadamente.

0	1	0
1	-4	1
0	1	0

• Fig. 4-10: Máscara utilizada para realizar la transformada de Laplace discreta en un imagen (detección de bordes)

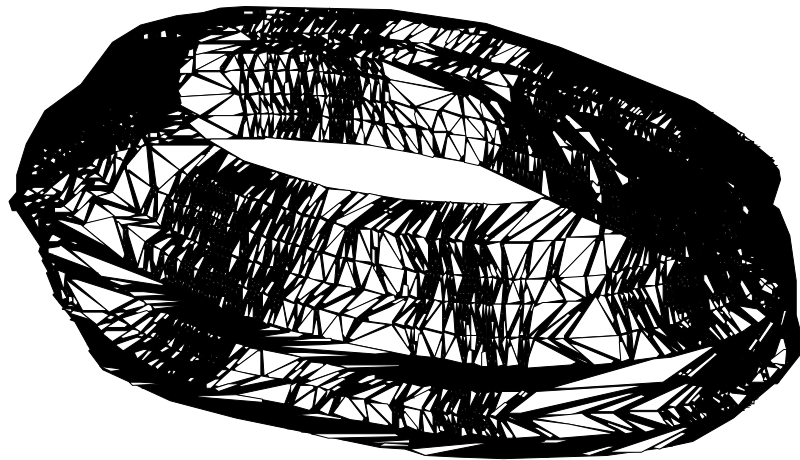
3. Se barre la imagen hasta encontrar un pixel marcado como frontera, entonces se invoca un seguidor de contornos (de 8 direcciones) que va "siguiendo" los pixels marcados como frontera y va guardando los pasos necesarios hasta volver al punto de partida.

4. Se procesa esa secuencia de pasos transformandose en rectas (vectores) con distintos X,Y e igual Z para un mismo nivel (para un mismo corte transversal).
5. Con la intención de reducir la cantidad de datos a procesar, se corre un rectificador de contornos que elimina aquellos de area insignificante (probablemente producto de errores de medición o adquisición), reduce el número de vértices y ordena los contornos de manera uniforme en sentido horario.

4.3.2.2 Parches Triangulares

La representación por parches triangulares es útil debido a la rapidez con la que puede ser mostrada en pantalla. Si bién la resolución obtenida no es igual a la de superficies B-Spline, permite un manejo interactivo del órgano, ya que el tiempo de respuesta del rendering de un modelo wireframe basado en parches triangulares está muy optimizado y ha sido estudiado largamente en el área de Computer Graphics.

En la (Figura 4-11) se puede observar la superficie externa del cráneo representada mediante parches triangulares, según lo produce nuestro sistema.



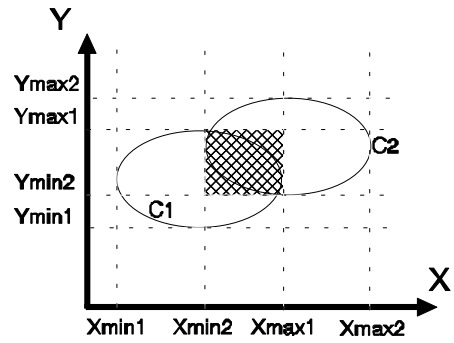
• Fig. 4-11: Representación de la superficie externa de la parte superior de un cráneo humano mediante parches triangulares

Para obtener esta representación, se parte de la estructura de contornos dirigidos nivelados recién explicada. El algoritmo de triangulación es una adaptación del presentado en [EKO91], que a su vez es una extensión de [CHR788], [BOI85-1] y [BOI85-2]. En líneas generales, tiene los siguientes pasos:

4.3.2.2.1 Algoritmo de Decisión de Encadenamiento

Se tiene un número n de contornos en el nivel k (corte k) y un número s de contornos en el nivel $k+1$ (corte $k+1$).

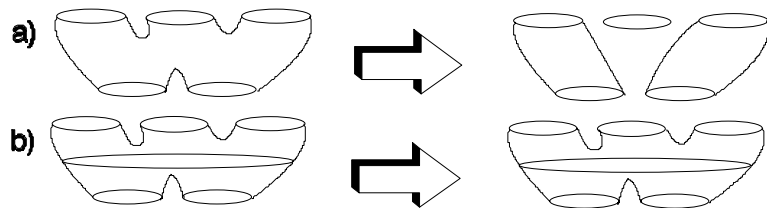
La primer tarea es decidir que contorno(s) del nivel k se debe encadenar con que contorno(s) del nivel $k+1$. Se adopta la siguiente regla heurística: se calculan las áreas de solapamiento (overlapping) (Figura 4-12) entre cada uno de los contornos del nivel k con cada uno de los contornos del nivel $k+1$.



• Fig. 4-12: Área de solapamiento entre dos contornos cerrados C1 y C2.

Este solapamiento se traduce en un porcentaje que se compara con un "umbral de certeza" (de 0 a 1) provisto por el usuario. Mientras mayor es el umbral de certeza mayor exactitud se pretende en las conexiones, pero mayor es también el riesgo de obviar una conexión necesaria que no superó el umbral quizás por errores en la adquisición de datos o en la registración de los distintos cortes. Es decir, se debe tomar una desición de compromiso balanceando la calidad que se pretende en la desición, con la calidad del equipamiento de pre-procesamiento que se dispone.

Un punto importante es la densidad de cortes. Si no se toma la cantidad de cortes necesaria se puede incurrir en el error de conectar erróneamente los contornos, como se aprecia en la (Figura 4-13).



• Fig. 4-13: (a) Ejemplo de SubSampling en la cantidad de cortes (b) Sampling correcto, las conexiones se hacen bien

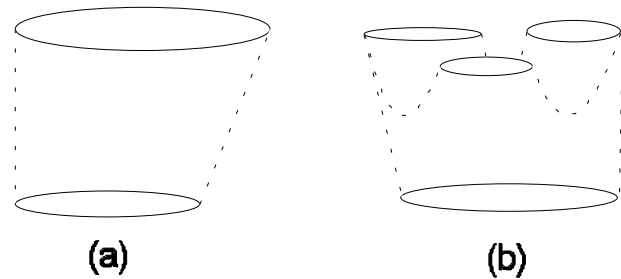
A medida que se va tomando la desición de cómo realizar el encadenamiento se va formando una matriz de conexiones. (Figura 4-14). Al terminar el proceso de desición, esta matriz tiene todas las conexiones que hay que hacer entre los dos cortes k y $k+1$.

	$C(k+1,1)$	$C(k+1, 2)$	...	$C(k+1, s)$		ΣC_k	Conexión
$C(k, 1)$	1	0	...	1		2	Multiple branching del cont.1 de k, con 1 y s de k+1
$C(k, 2)$	0	1	...	0		1	Single Branching cont. 2 de k, con cont. 2 de k+1
...	...	...	...				
$C(k, n)$	0	0	...	1		1	Single Branching cont. n de k, y s de k+1
$\Sigma C(k+1)$	1	1		2			

• Fig. 4-14: Matriz de conexiones entre C_k y C_{k+1}

Los tipos de conexiones posibles (mostrados en Figura 4-15) son:

- Uno a Uno (Single-Branching). Cuando un contorno de un nivel se encadena a uno y solo uno de otro nivel.
- Uno a Muchos / Muchos a Uno. (Multiple-Branching) Cuando en un nivel existe un contorno "abarcador" de varios en otro nivel.



• Fig. 4-15: (a) Encadenamiento uno a uno (single branching) (b) Encadenamiento uno a muchos (multiple branching)

El método de generación de triángulos en el caso de Multiple-Branching es una extensión del caso general de Single-Branching, que se esboza a continuación:

4.3.2.2.2 Single Branching entre Contornos Convexos

Dados $C(k)$ y $C(k+1)$, dos contornos dirigidos cerrados coplanares, en el corte k y $k+1$ respectivamente, generados uniformemente mediante una lista de vértices, la generación de parches triangulares que cubran (o interpolen) la superficie determinada por el promedio de la extrusión de ambos contornos a lo largo de un eje perpendicular al plano sobre el cual están inscriptos, se define como un secuencia de pasos, a saber:

Paso A: Siendo $nv(k)$ y $nv(k+1)$ el número de vértices de $C(k)$ y $C(k+1)$ respectivamente, se define (Figura 4-16):

$$nvMAX = \max[nv(k), nv(k+1)]$$

$$nvMIN = \min[nv(k), nv(k+1)]$$

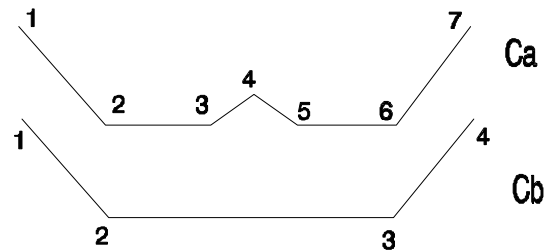
$$Ca = C(k), Cb = C(k+1) \text{ si } nvMAX = nv(k)$$

o también

$$Ca = C(k+1), Cb = C(k) \text{ si } nvMAX = nv(k+1)$$

donde

Ca es el contorno con mayor número de vértices



• Fig. 4-16: Dos segmentos de "contornos cerrados". Para la figura, $nvMAX = 7$ y $nvMIN = 4$, siendo Ca y Cb los mostrados.

Paso B: Para cada vértice de Cb se busca el vértice más "cercano" en Ca, manteniendo la localidad y sin generar cruzamientos. Se va generando una lista de bordes (BL) que representa las conexiones de cada vértice de Cb (campo vértice1) con alguno de Ca (campo vértice2).

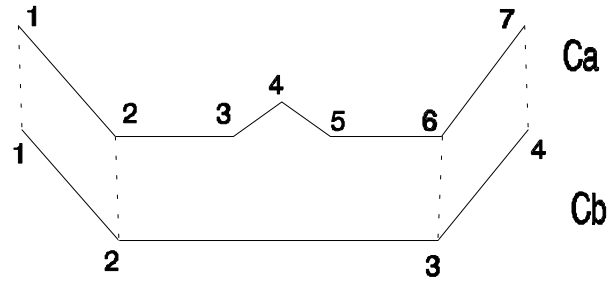
```

UltimoEncadenado = 0
for i = 1 to nvMIN
    BL[i].vértice1 = Cb[i]
    BL[i].vértice2 = Ca.Cercano(Cb[i], UltimoEncadenado)
    UltimoEncadenado = BL[i].vértice2
end

```

El miembro Cercano recibe un vértice y un número de vértice y devuelve un vértice (del contorno del cual es miembro) con la característica de ser el más próximo al vértice enviado como parámetro. El parámetro UltimoEncadenado se utiliza en Cercano para mantener la localidad (no ir hacia atrás y generar cruzamientos). Para el cálculo de la distancia se utiliza la fórmula de distancia Euclídeana (en R2), ya que la distancia "aportada" por la coordenada Z es uniforme para todas las comparaciones y puede entonces, eliminarse, optimizando así el cálculo.

Luego de este paso, aplicado a la (Figura 4-16), tendríamos una situación como la mostrada en (Figura 4-17) y una tabla de bordes como (Figura 4-18).



• Fig. 4-17: Situación de (4-16) luego de encadenados los vértices mas próximos entre sí. (bordes en líneas punteadas)

vértice1	vértice2
1	1
2	2
3	6
4	7
-	-
-	-
-	-

• Fig. 4-18: Situación de BL (lista de bordes), luego de los pasos A y B

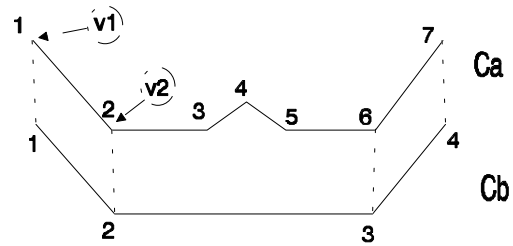
Paso C: Se procesan, uno a uno, los vértices no encadenados de Ca, uniéndolos a los más próximos en Cb, de la siguiente manera:

```

Agregados = 0
for i = 1 to nvMIN                /* cantidad de bordes en BL */
    v1 = BL[i].vértice2
    v2 = BL[i+1].vértice2
    if (v2 - v1) > 1                /* si hay vértices entre los bordes ya */
        for j = 1 to (v2 - v1) /* generados i e i+1 los proceso */
            Agregados ++
            d1 = Distancia(Cb[BL[i].vértice1], Ca[v1+j])
            d2 = Distancia(Cb[BL[i+1].vértice1], Ca[v1+j])
            if d1 > d2
                BL[nvMIN+Agregados].vértice1 = BL[i+1].vértice1
                BL[nvMIN+Agregados].vértice2 = v1+j
            else
                BL[nvMIN+Agregados].vértice1 = BL[i].vértice1
                BL[nvMIN+Agregados].vértice2 = v1+j
            end
        end
    end
end
end
end
end

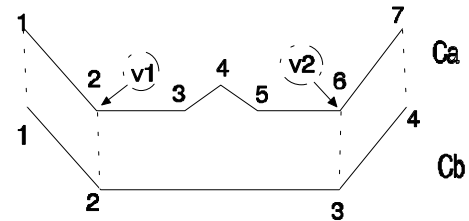
```

En la primera iteración del loop mas externo (en i), v1=1 y v2=2, o sea, no hay vértices en el medio, y se continua el loop (Figura 4-19).



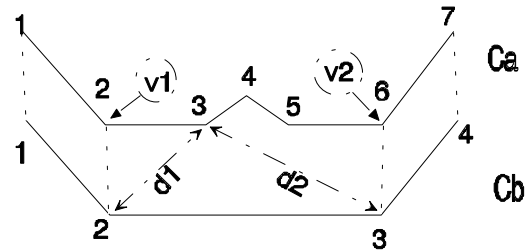
• Fig. 4-19: v1 y v2 luego de la primera iteración (en i)

En la segunda iteración, en cambio, $v1=2$ y $v2=6$, y $(v2-v1) > 1$, o sea, existen vértices entre v1 y v2 que no han sido conectados (Figura 4-20).



• Fig. 4-20: v1 y v2 luego de la segunda iteración en i

El loop sobre j procesa, uno por uno, los vértices de Ca que están entre v1 y v2. Para cada uno, se calcula (función Distancia) la distancia a los vértices de Cb conectados a v1 y v2 respectivamente, eligiéndose el mas cercano de estos para hacer la conexión. Se selecciona el mas cercano para no formar, en lo posible, triangulos largos y finos, que no se ven bien en el rendering. En la (Figura 4-21) tenemos la situación luego de la primer iteración en j.



• Fig. 4-21: Primera iteración sobre j. Como $d1 < d2$, se conectaría Ca(3) con Cb(2)

vértice1	vértice2
1	1
2	2
2	3
2	4
3	5
3	6
4	7

• Fig. 4-22: Lista de bordes completa y ordenada según vértice2

Paso D: En este momento, tenemos la lista de bordes BL completa con todas las uniones que podemos hacer. Se ordena esta lista según el campo vértice2 (Figura 4-22) y se van procesando los bordes de a pares, generando los triángulos a partir de ellos siempre en sentido horario y siguiendo ciertas convenciones para permitir una mas fácil graficación, de la siguiente manera:

Siendo i y j los índices (dentro de BL) de un par de bordes consecutivos, de manera tal que $j = i + 1$:

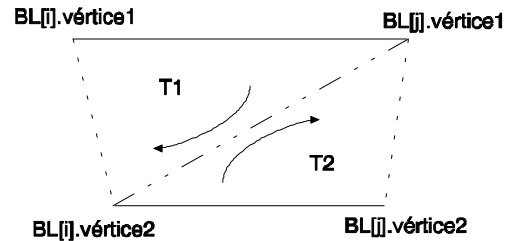
```

if (BL[i].vértice1 <> BL[j].vértice2) and
   (BL[j].vértice1 <> BL[i].vértice2) then
    (* caso 1. cuatro vértices *)
    (* distintos *)
    Triángulo(BL[i].vértice1, BL[j].vértice1, BL[i].vértice2)
    Triángulo(BL[j].vértice1, BL[j].vértice2, BL[i].vértice2)
else
    (* caso 2. uno de los vértices se repite *)
    ... busco el vértice que se repite ...
    ... y triangulo con los otros tres ...
end

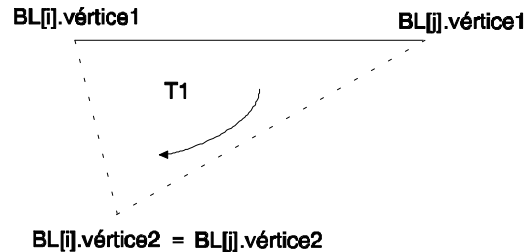
```

Al procesar la lista de bordes tenemos dos casos posibles. Si los cuatro vértices son distintos, la triangulación es simple, generándose dos triángulos con la disposición espacial mostrada en la (Figura 4-23). Este caso se daría, en el ejemplo que estamos siguiendo, entre los bordes 1 y 2, y 6 y 7.

Si los bordes coinciden en algún vértice, se identifica este y se genera el único triángulo posible (Figura 4-24). En el ejemplo, los pares de bordes 2 y 3, 3 y 4, 4 y 5, y 5 y 6 serían ejemplos de este caso.

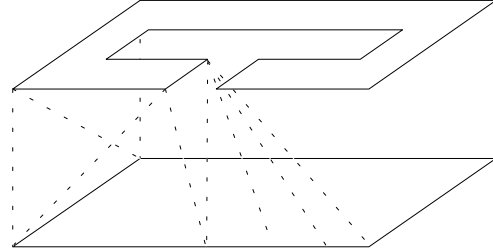


• Fig. 4-23: Triangulación de bordes sin vértices en común. Se generan los triángulos T1 y T2, con el sentido mostrado por las flechas.



- Fig. 4-24: Triangulación de bordes con un vértice en común

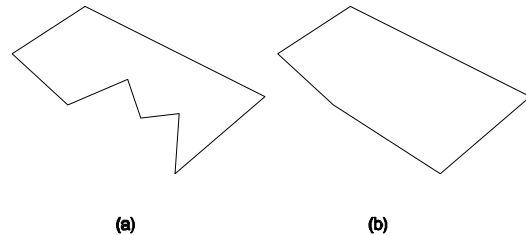
Para solucionar los problemas que presenta el método de Christiansen [CHR78} (Figura 4-25) al triangular dos contornos con siluetas disímiles, se transforman todos los contornos al máximo contorno convexo posible (convex-hull).



- Fig. 4-25: Resultados incorrectos del algoritmo de Christiansen al intentar encadenar dos contornos con siluetas distintas

4.3.2.2.3 Obtención de la Cápsula Convexa (Convex-Hull)

El convex-hull de un polígono es el máximo polígono convexo que lo contiene (Figura 4-26).



- Fig. 4-26: (a) contorno cerrado cualquiera (b) convex-hull

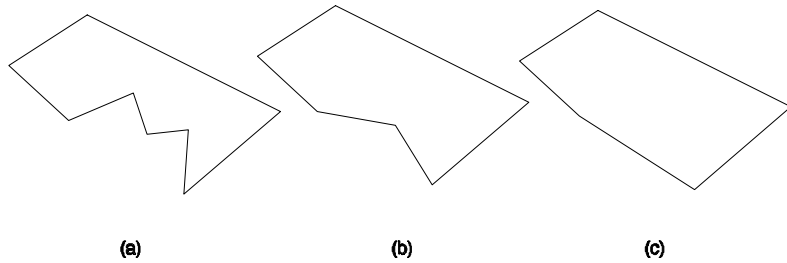
El algoritmo para obtener el convex hull de cualquier polígono es el siguiente:

```

loop
  listo = TRUE
  loop (i)
    if (Concavidad(vértice(i), vértice(i+1), vértice(i+2)) = TRUE)
      listo = FALSE
      borro vértice(i+1)
    end
  until (i = número_de_vértices_en_polígono)
until (listo)

```

En cada iteración, el algoritmo reduce (borra) todas las concavidades de primer orden, transformando por consiguiente las de 2do. en 1ero, etc. Eventualmente, no existirán mas concavidades y el polígono resultante es el convex-hull. La (Figura 4-27) ejemplifica la secuencia de iteraciones y sus resultados al aplicar el algoritmo al polígono de la (Figura 4-26).



• Fig. 4-27: (a) Polígono original, (b) Concavidades de primer orden removidas, (c) Concavidad de 2do orden removida = convex-hull

La función Concavidad recibe tres puntos consecutivos y determina, basandose en una orientación definida como global para todo el sistema, si forman concavidad o nó.

Si es así, se elimina el vértice que genera la concavidad (el del medio) y se continúa procesando.

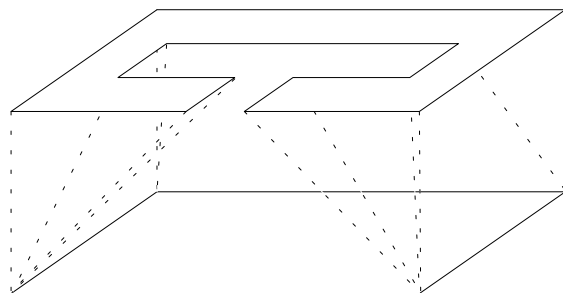
4.3.2.2.4 Secuencia Final de Pasos

El procedimiento para encadenar dos contornos cualquiera, sería entonces:

- Obtener los convex-hull de ambos contornos.
- Hacer el single-branching entre estos.
- Basándonos en las uniones realizadas (haciendo analogía, por ejemplo, con el número de vértice de las conexiones), encadenar de la misma forma los contornos originales.

La idea es usar los maximos contornos convexos para hacer el encadenamiento, ya que ambos, por la definición del algoritmo de desición para realizar encadenamiento (4.3.2.2.1), deberían tener formas similares. Obtenidas así las uniones (en términos de números de vértices y nó en coordenadas espaciales absolutas), se hace la analogía con los contornos originales y son estos, en definitiva, con los que se forman los triángulos.

Con este procedimiento, la (Figura 4-25) se transforma en (Figura 4-28).



- Fig. 4-28: Resultados del algoritmo de triangulación aplicado a los contornos "problemáticos" (solo se muestran los bordes)

4.3.2.2.5 Multiple Branching

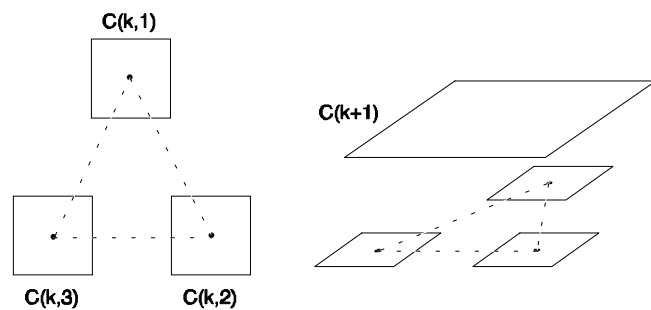
Cuando un objeto tiene forma compleja, puede darse la situación mostrada en la (Figura 4-15b). En ese caso, es necesario encadenar un contorno de un nivel con varios de otro.

El procedimiento para lograr el encadenamiento se basa en lo visto anteriormente y consiste en la siguiente secuencia:

Dados $C_{k,1}, C_{k,2}, \dots, C_{k,n}$; contornos en el nivel k ; y C_{k+1} ; contorno en el nivel $k+1$, el multiple branching entre los contornos de k , con los de $k+1$ se define como:

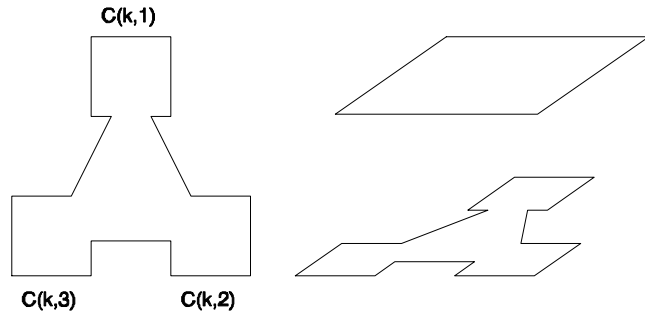
1. Se genera un único contorno uniendo los centroides de $C_{k,1}, C_{k,2}, \dots, C_{k,n}$
2. Este contorno se encadena (sin generar triángulos) con C_{k+1} . Se guarda la lista de bordes obtenida.
3. Se genera un contorno interpolado, C' , entre los niveles k y $k+1$, usando la intersección de un plano coplanar con k y $k+1$ (pero en la mitad del camino entre los niveles k y $k+1$), con los bordes generados en el paso 2.
4. Se encadena C' con C_{k+1} , generandose los triángulos.
5. Para cada contorno del nivel k ($1 \dots n$), se encadena con su homólogo en C' , uniendo los vértices de $C_{k,i}$ que no formen parte de C' , con los vértices que sí lo hagan.
6. Se triangula planarmente en el nivel $Z = (z(k) + z(k+1))/2$ para "cerrar" la superficie.

Como ejemplo, supongamos que tenemos tres contornos en un nivel k que debemos encadenar con un único contorno en un nivel $k+1$ (Figura 4-29).



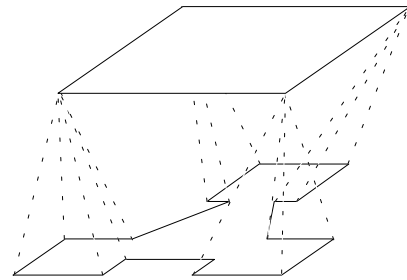
- Fig. 4-29: Vista de planta de contornos unidos por sus centroides y vista 3D de la ubicación de todo el sistema.

Al unir los centroides de los tres contornos obtenemos el "contorno abarcador" mostrado en (Figura 4-30). Notesé que algunos vértices de los contornos que forman este "contorno abarcador" no forman parte de este último.



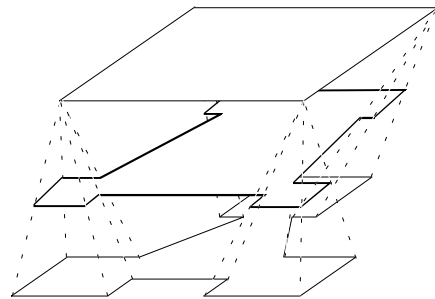
• Fig. 4-30: Contorno formado por la union de los tres contornos individuales en el nivel k.

Realizamos el single branching entre el "contorno abarcador" y el contorno único del nivel $k+1$, pero no generamos los triángulos respectivos, sino que guardamos la tabla de bordes. En la (Figura 4-31) se observan los bordes generados luego de realizado este paso.



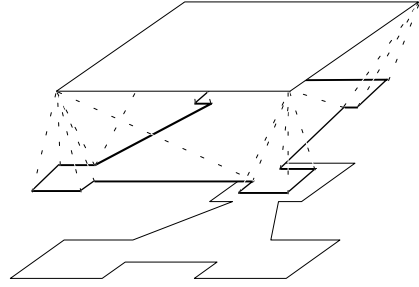
• Fig. 4-31: Single-Branching entre el contorno unión de los n contornos del nivel k y el contorno del nivel $k+1$ (se muestran los bordes)

A un nivel intermedio entre los niveles k y $k+1$ formamos el contorno interpolado C' . Los vértices de este son, justamente, la intersección de c/u de los bordes generados en el paso anterior con un plano coplanar al plano que contiene los contornos en cuestión, pero al nivel $Z = (Z(k) + Z(k+1)) / 2$. (Figura 4-32).



• Fig. 4-32: Contorno interpolado C' , en el nivel $Z = (z(k)+z(k+1))/2$ (en línea gruesa)

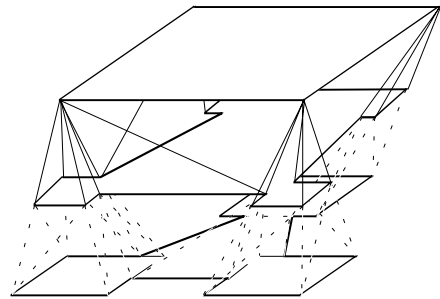
Una vez generado el contorno C' , hacemos el single branching entre este y el único contorno del nivel $k+1$. Esta vez, sin embargo, sí generamos los triángulos correspondientes. (Figura 4-33)



• Fig. 4-33: Single Branching entre C' y contorno de $k+1$. Se muestran los triángulos formados (líneas de puntos)

Debemos ahora, conectar cada contorno individual del nivel k con C' . Unimos entonces, los vértices de los contornos $C(k,..)$ con sus proyecciones en C' . Pero con aquellos vértices que no formaron C' , debemos tener una consideración especial.

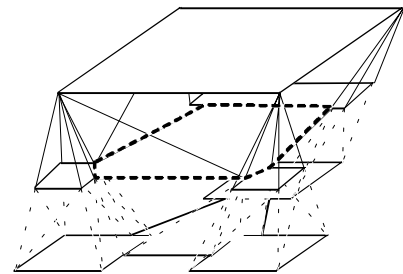
Unimos cada uno de estos vértices con el vértice proyectado (sobre C') mas próximo. Si estamos sobre el último de estos vértices, debemos conectarlo a ambos extremos "proyectados" en C' (ver Figura 4-34).



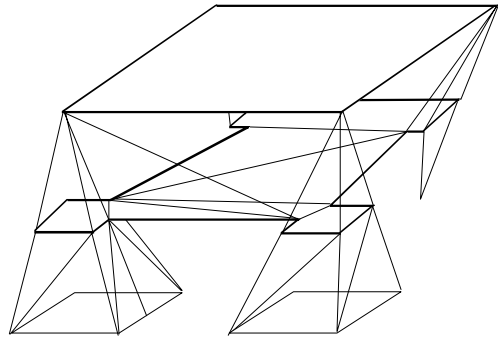
• Fig. 4-34: Triángulos de (4-33) con líneas sólidas. Single branching entre c/u de los contornos del nivel k con sus homologos en C' (en líneas de puntos)

Finalmente, debemos triangular en el plano de C' , la zona que queda abierta para poder, así, cerrar el volumen. La zona a cerrar es la resta booleana de C' menos todos los contornos del nivel k . En la (Figura 4-35) se observa esta en línea de puntos.

El resultado de todo el proceso, a manera de un simple esquema, se muestra en la (Figura 4-36).



- Fig. 4-35: Zona a triangular (en el plano) para cerrar la superficie (línea de trazos gruesa)



- Fig. 4-36: Resultado final de la triangulación. (solo se han mantenido los triangulos inmediateamente visibles para hacer el esquema mas simple)

CAPITULO V

VISUALIZACION

§ 5.1 Introducción

Por visualización se entiende todo lo relacionado directamente con la presentación de los datos en pantalla.

Si bien el presente trabajo no pretende ser referencia de implementaciones de visualización, interesó desde el primer momento concentrar la atención en la utilidad y la posibilidad de extensión en la investigación llevada a cabo hacia diferentes métodos y/o entornos, más que en la belleza estética propiamente dicha en las presentaciones.

Dentro de la visualización de volúmenes podemos identificar, como se mencionó anteriormente, dos tendencias: volume rendering y surface rendering.

Estos distintos acercamientos al problema de la representación gráfica de volúmenes tridimensionales tienen métodos diferentes y se justifica, así, estudiarlos por separado.

§ 5.2 Volume Rendering

La visualización por medio de volume rendering se basa en la información "bruta" que posee el espacio de datos. Esto es, no se intenta hacer una representación por medio de superficies, líneas o algún otro método, sino que el único elemento de datos es el voxel mismo.

Luego de la identificación del órgano de interés (4.2), el espacio de datos queda reducido a una escena binaria (binary scene) donde cada voxel tiene uno de dos valores posibles, indicando si el mismo pertenece, o nó, al objeto siendo estudiado.

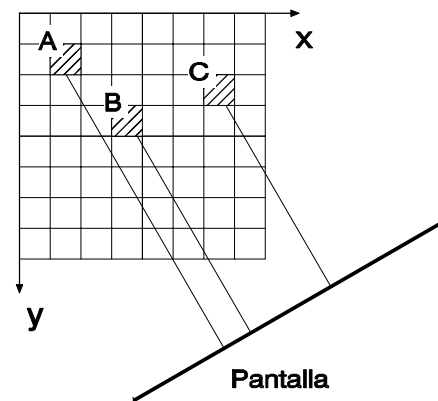
Existen diversos métodos para representar el espacio de datos de la escena binaria. H. Samet [SAM90-1] [SAM90-2] y S. Tanimoto [TAN80] presentan las estructuras de octrees y pirámides, respectivamente. Se recomienda al lector interesado remitirse a estos trabajos para un tratamiento en detalle.

Aunque trabajar en volume rendering requiere de equipos poderosos para lograr resultados aceptables, fué de nuestro interés estudiar, aunque mas no sea de manera experimental, algun método de graficación basado en este tipo de representaciones.

A continuación se presentan un método simple de visualización de volumen, a través de un recorrido atras-adelante del espacio de datos, y un método híbrido de representación (y visualización) que ha sido utilizado largamente por los especialistas en el area de medical imaging.

5.2.1 Método Back-To-Front para Visualización de Volúmenes

Este método [FRI85] gráfica una colección de voxel cúbicos con un mínimo de preprocesamiento y sin computar las superficies del objeto. Por esta razón, es aplicable en equipos de moderada capacidad.



• Fig. 5-1: Disposición supuesta del plano de la pantalla en relación a un espacio de datos bidimensional con tres pixels "encendidos"

La (Figura 5-1) muestra una supuesta disposición de un espacio de datos bidimensional (la extensión a 3D es directa) con tres pixels "OBJETO" y el plano de la pantalla.

El algoritmo BTF (back-to-front) recorre secuencialmente, uno por uno, todos los pixels (voxels) de la imagen (escena binaria), proyectando cada uno en la pantalla. El secreto para generar una imagen coherente está en el orden del recorrido del espacio de datos.

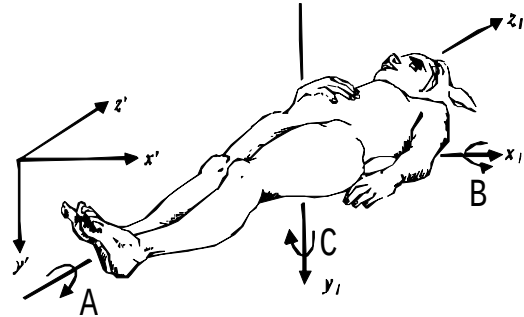
Tomando como ejemplo la (Figura 5-1), si el orden de barrido es A, B, C, tenemos que, al proyectar el pixel A, este ocupa (en la pantalla) la posición que debería ocupar el pixel C, por estar este último, mas "cerca" del observador. Sin embargo, cuando llegamos al pixel C, también proyectamos este, sobrescribiendo (en la pantalla), lo que hubiera y dándole el valor de intensidad de C (o, según el método de sombreado que utilizemos, un valor acorde con la distancia pantalla-pixel C).

Si el orden de barrido, en cambio, fuera tal que C sea proyectada antes que A, se generaría un error en la imagen, ya que pixels mas "lejanos" al observador obstruirían a pixels mas "cercanos".

El orden de barrido, entonces, es el punto mas importante de este método, que se podría resumir como: "Dados un plano de proyección (pantalla) y un espacio de datos, encontrar el orden de barrido apropiado, de manera tal que los elementos mas lejanos al plano de

proyección sean procesados siempre antes que los elementos relativamente mas cercanos"

Nuestro espacio de datos, como viene de los equipos TAC, tiene la disposición espacial mostrada en (Figura 5-2).



• Fig. 5-2: Ubicación espacial del espacio de datos del TAC. Notesé que los cortes se realizan coplanares con el plano x',y'

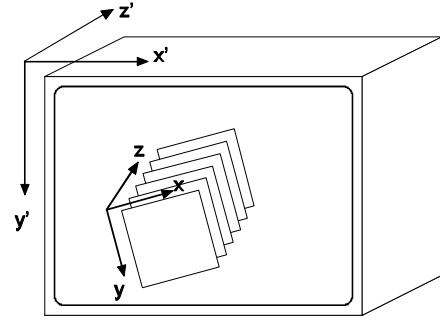
Definido este sistema coordenado (object space), determinamos ahora un sistema coordenado para la imagen (image space). Definimos este con los ejes x e y coplanares con la pantalla, y con el eje z creciente a medida que se "aleja" del observador.

Tenemos, entonces, un espacio de datos definido sobre un sistema coordenado como se muestra en (Figura 5-2). Para obtener distintas vistas de este espacio, utilizamos los ángulos A , B , y C para rotar el volumen.

Para definir un orden correcto de barrido de los voxels, transformamos este espacio de datos hasta que su sistema coordenado (ya rotado según el punto de vista deseado) coincida con el espacio de la imagen. Una vez obtenido esto, la secuencia de barrido es siempre la misma, como sigue:

1. Loop en la dirección Z (slice por slice)
2. Dentro de cada slice, loop sobre las filas (dirección Y)
3. Loop sobre la dirección X
4. Proyectar el voxel(x,y,z) y colorear

La relación entre los espacios de objeto (object space) y de imagen (image space) se muestra en (Figura 5-3).



• Fig. 5-3: Espacio de objeto (x,y,z) y de imagen (x',y',z')

La transformación object-space --> image-space es la siguiente:

Dado un voxel con coordenadas (x,y,z) en el espacio de objeto y los ángulos de rotación A, B, y C; las coordenadas en el espacio de imagen (x',y',z') están dadas por:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = T \cdot \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix}$$

donde:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

y T es la matriz de transformación, que se obtiene:

$$T = \begin{bmatrix} t_{ij} \end{bmatrix}$$

$$= S \cdot \begin{bmatrix} \cos A \cdot \cos C + \sin A \cdot \sin B \cdot \sin C & -\sin A \cdot \cos C + \cos A \cdot \sin B \cdot \sin C & \cos B \cdot \sin C \\ \sin A \cdot \cos B & \cos A \cdot \cos B & -\sin B \\ -\cos A \cdot \sin C + \sin A \cdot \sin B \cdot \cos C & \sin A \cdot \sin C + \cos A \cdot \sin B \cdot \cos C & \cos B \cdot \cos C \end{bmatrix}$$

Siendo S el factor de escalamiento $\left(\frac{1}{\sqrt{3}} \right)$ para orientaciones arbitrarias, y (x_c, y_c, z_c) el centro del volumen de datos (object space), y (x_s, y_s, z_s) el centro del (image space). Siendo normalmente (x_s, y_s) el centro de la pantalla y z_s la cantidad de colores dividido 2.

Una forma de permitir factores de escala arbitrarios es reemplazar cada voxel por un esfera cuya proyección es un disco de diámetro $\sqrt{3}$, donde a es el tamaño del voxel luego aplicarse el factor de escala deseado.

Otra forma de no perder resolución la realizar transformaciones con orientaciones arbitrarias es reemplazar cada voxel por una sucesión de superficies cuadradas (ver 5.2).

Para concluir el método, es importante hacer notar que pueden obtenerse vistas parciales (de subvolumenes) simplemente cambiando los límites de los loops en el corazón del algoritmo.

5.2.2 Método Híbrido

El método recién descripto posee inconvenientes de pérdida de resolución cuando el volumen de datos de rota a una posición arbitraria no paralela con ninguno de los ejes. Además, la manejabilidad de el espacio de datos es cuestionable, ya que no podemos modificar, de manera simple, los órganos que allí se representan.

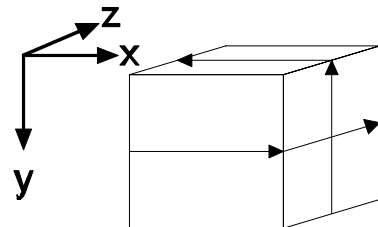
Investigadores del Medical Image Processing Group, de la Universidad de Pennsylvania, liderados por el Dr. Herman y J.K Udupa, propusieron, entonces, un método híbrido para representar estructuras espaciales.

Este método se denomina "híbrido" porque amalgama la representación biunívoca de cada voxel, pero lo representa por medio de entidades matemáticas.

Para llegar a la representación final, primero se genera la escena binaria (mediante los métodos de segmentación vistos anteriormente).

Luego, se ejecuta un procedimiento de seguimiento de superficie (surface tracking) que va recorriendo la escena binaria y devuelve un conjunto de superficies planas cuadradas que representan la superficie externa del órgano dentro de la escena.

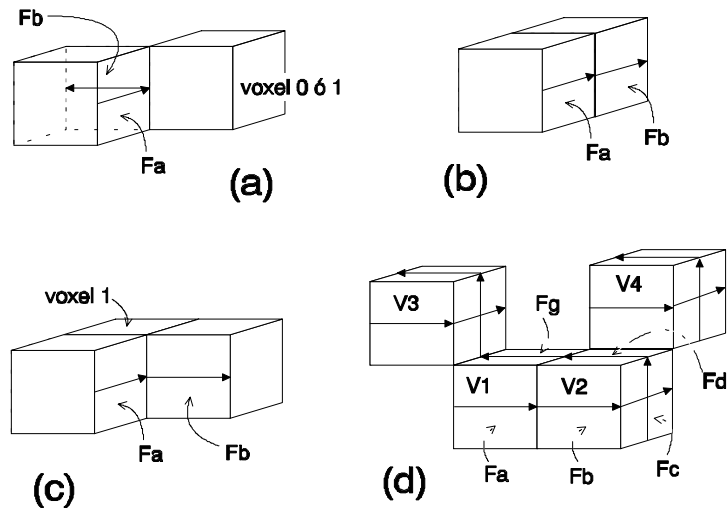
Las superficies se van generando una por una, a medida que se recorre el volumen de datos. Para mantener la coherencia en la formación de las superficies, se definen "circuitos" en cada voxel (Figura 5-4).



• Fig. 5-4: Voxel con los circuitos definidos para determinar la conectividad de objetos y superficies

Un objeto se forma conectando voxels unos con otros. Se usa una relación de conectividad que permite la conectividad sólo por donde pasan los circuitos (que, en realidad, son todas las caras del voxel). El objeto es, entonces, el máximo conjunto conexo de superficies.

El algoritmo empieza en un voxel cualquiera (encontrado mediante un barrido de afuera hacia adentro) y se empiezan a seguir los circuitos. Se debe definir la adyacencia de un voxel con otro. En la (Figura 5-5) se observan distintos objetos y como se comporta el algoritmo para distintos tipos de proximidad de voxels.



- Fig. 5-5: Ilustración de la definición de conectividad de caras. (a) Fa es adyacente a Fb, el voxel mostrado -conectado por un borde solamente- se ignora; (b) ambos voxels conectados por una cara y sus superficies adyacentes; (c) situación similar a (a), pero ahora existe un voxel de respaldo, se generan las superficies Fa y Fb; (d) en este ejemplo, se identificarían tres objetos distintos: $O1 = V3$, $O2 = V4$ y $O3 = \{V1, V2\}$. Nota: excepto en (a) -y se aclara-, en todos los casos se muestran solo los voxels "1" (pertenecientes al objeto)

De esta manera, se forman los conjuntos de superficies, que luego se grafican utilizando los algoritmos estándares de computación gráfica.

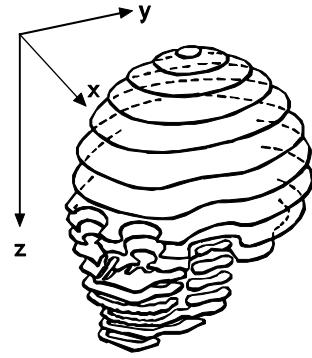
§ 5.3 Surface Rendering

Los métodos de graficación de superficies tridimensionales han sido estudiados extensamente en computación gráfica, existiendo excelente literatura sobre el tema de graficación en general [FOL90], y aplicada a estructuras espaciales de datos en particular [SAM90-2].

Nos limitamos, entonces, a explicar como organizamos en nuestro sistema, los parámetros de visualización.

5.3.1 Contornos Dirigidos Nivelados

Basándonos en la disposición espacial de los cortes del equipo de TAC (Figura 5-2), tenemos que los contornos nivelados (generados en 4.3.2.1) tendrían el sistema coordenado mostrado en (Figura 5-6).



• Fig. 5-6: Sistema coordenado de los contornos dirigidos nivelados generados por nuestro método

Es decir, tenemos un número indefinido de contornos en cada corte que recibimos del equipo TAC. Todos los contornos de cada corte tienen la misma coordenada Z y, debido al método utilizado para obtenerlos, están definidos en sentido horario, tomando como primer vértice al que posea menor x,y (esquina superior izquierda de la pantalla).

Ya definido el sistema coordenado de los contornos, la graficación de los mismos se logra con una simple transformación como la vista en 5.2.1.

5.3.2 Parches Triangulares

Los parches triangulares se forman a partir de los contornos dirigidos nivelados. Por este motivo, comparten el sistema de coordenadas con estos.

Sin embargo, para facilitar la graficación, los triángulos deben formarse siguiendo ciertas convenciones. Las siguientes son las que nosotros utilizamos:

- Los triángulos se definen en sentido horario
- El primer vértice del triángulo es el de mayor (o menor) Z, Notesé que no importa cual de ambas convenciones elijamos, sinó que lo importante es seguir siempre la misma convención.
- Las normales a la superficie de los triángulos apuntan siempre hacia "afuera" del objeto que los triángulos pretenden representar. (Esto es una consecuencia de los puntos anteriores)

§ 5.4 Sombreado (Shading)

El sombreado artificial de imágenes es un area de mucha investigación y existen numerosos trabajos sobre el tema [FOL90] [SAM90-2] [NOV90] [DUF79].

En los algoritmos estándares de sombreado, existen dos puntos principales que hay que resolver:

- ¿Qué? estructura de datos estamos utilizando y ¿cómo? calcular el color apropiado para alguno de sus puntos.
- ¿Cómo? pintar la superficie una vez determinado un color

El primer aspecto se refiere a determinar, según la posición en el espacio, las luces, sombras, textura del objeto, y otros condicionantes apropiados un color para el objeto en cuestión.

Un método muy usado por su simplicidad es hacer una relación directa entre la intensidad del punto y su distancia al observador. Es importante hacer notar que esta distancia es muy simple de obtener, ya que es la coordenada Z, una vez que el punto ha sido transformado del object space al image space.

El segundo aspecto se refiere a como distribuir el color elegido dentro del objeto (triángulo, cuadrado, etc.). La forma más simple es pintar todo el objeto del mismo color (constant shading), pero esto se traduce en diferencias notables entre un objeto y los adyacentes.

Otro método mas interesante es calcular no un color para el objeto, sino varios (en diferentes puntos). Luego se subdivide el objeto en elementos mas pequeños y se hace un promedio de los colores elegidos pintándose cada elemento del mismo color (interpolated shading). Manejando la cantidad de divisiones en los objetos se controla el detalle de la imagen.

Por último, la mejor forma de obtener imágenes realistas es utilizar la técnica denominada seguimiento de rayos (ray tracing). Este método simula la trayectoria de cada uno de los rayos de luz que llegan a cada pixel de la pantalla, simulando los choques con los distintos objetos del object space y, eventualmente, con alguna fuente de iluminación.

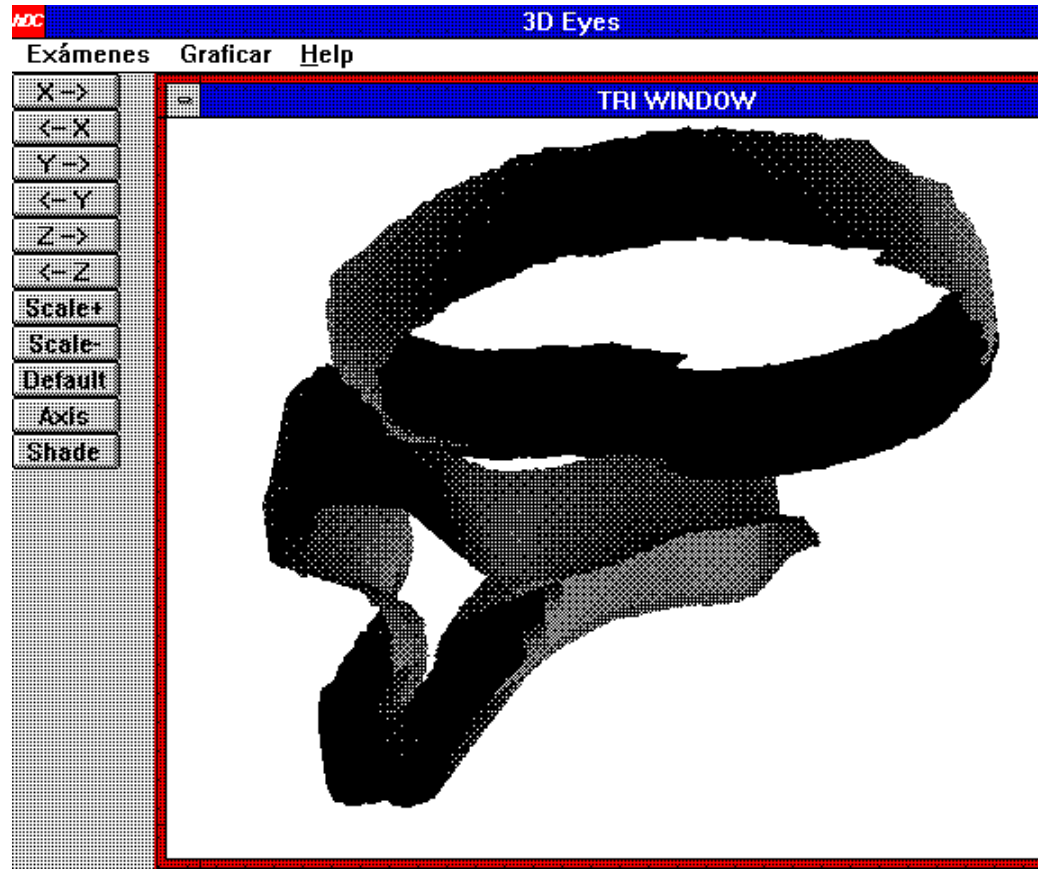
Por su alto costo computacional, el método de ray tracing no es apropiado para el diseño de un sistema interactivo, pero sí se utiliza para la generación de fotos finales, animación computada y realismo de imágenes.

5.4.1 Nuestra Implementación

Utilizamos el algoritmo de interpolated shading. Podemos resumirlo en los siguientes pasos:

1. Calculamos el punto medio de cada triángulo
2. Los ordenamos según este punto, de "mas lejano" a "mas cercano"
3. Loop a traves de los triángulos ordenados
4. Calculamos los colores (según la coordenada Z) de cada vértice del triángulo
5. Subdividimos el triángulo en N triángulitos mas pequeños
6. Pintamos estos triangulitos según el gradiente de los colores obtenidos en (4)

En la (Figura 5-7) tenemos un ejemplo de los resultados de este algoritmo.



• Fig. 5-7: El sistema de graficación con una imagen de una vértebra sombreada utilizando color-por-distancia e interpolated shading

También investigamos con los algoritmos de ray tracing, pero el tiempo de respuesta era demasiado largo como para incluirlos en el sistema-de-usuario-final.

§ 5.5 Entorno GUI (Graphical User Interface)

El sistema interactivo de visualización es la parte esencial de todo el sistema. El resultado de todo el proceso se ve a través de estos "ojos". Es así que hay que poner especial cuidado en un aspecto crucial en el desarrollo de todo sistema interactivo: el tiempo de respuesta.

Otro de los objetivos fundamentales del trabajo en el análisis de decisión sobre la metodología a utilizar fué la facilidad de uso. Los posibles usuarios de un sistema como el propuesto serían personas sin ningún tipo de capacitación ni experiencia en informática. Por este motivo, se decidió implementar el sistema de manera tal que pueda ser utilizado como un juego.

Un entorno GUI es el ambiente adecuado para el sistema. Para desarrollar el prototipo utilizamos Microsoft Windows pero idénticos resultados pueden obtenerse utilizando X-Windows bajo plataformas Unix.

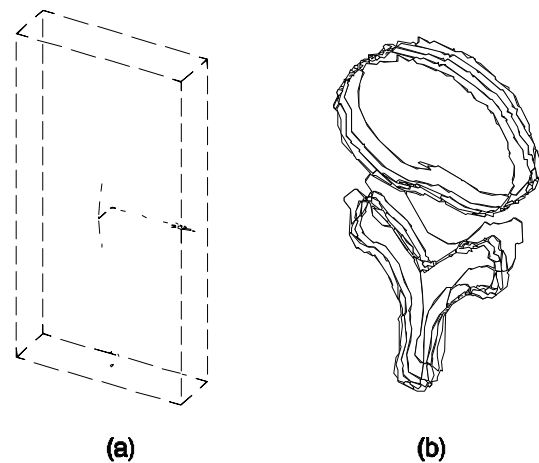
Como dispositivo de entrada usamos principalmente el mouse. Si bien este no es ideal para el ingreso de datos tridimensionales (ya que se maneja con coordenadas planares),

se puede diseñar, mediante software, modelos del espacio tridimensional para que el usuario ubique su posición exactamente en las tres dimensiones.

5.5.1 Modalidades de Visualización

El sistema de graficación debe permitir la modificación casi instantánea de los puntos de vista, hasta lograr la ubicación deseada, y la graficación completa (sombreado, coloreado) una vez lograda esta. Si bien la graficación de los contornos es rápida, al incrementar el número de cortes empieza a notarse el retardo en cada ciclo "rotación -> redibujo". Esto, sumado al hecho de que el usuario desea mover demasiado los puntos de vista hasta decidirse por alguno, motivó la realización de la caja límite (bounding box).

Cada estudio puede mostrarse como una caja del tamaño exacto (Figura 5-8) para contener todo el volumen con los ejes dibujados en distintos colores. La graficación de esta caja es instantánea y por lo tanto, el usuario puede rotarla y/o moverla hasta encontrar la ubicación deseada, cambiando entonces la modalidad de graficación de "Axis" (o ejes) a "Default" (por defecto).



• Fig. 5-8: (a) Representación por bounding-box o caja límite "Axis" (b) la imagen obtenida con contornos nivelados "Default"

Cada ventana del sistema puede mostrarse en una de tres modalidades:

- *DEFAULT*: Normal, se muestran los contornos dirigidos nivelados o los parches triangulares, según sea el archivo de input utilizado.
- *AXIS*: Se muestra la caja límite.
- *SHADE*: Si el archivo de entrada contiene parches triangulares, se observa una imagen sombreada del objeto.

La representación por DEFAULT está basada en estructuras de alambre (wireframe) y tiene por objeto obtener una vista rápida del objeto.

La rutina de SHADING se basa en un sombreado por interpolación entre los tres vértices del triángulo que está siendo sombreado. El color en cada vértice se obtiene por una relación directa con la distancia al observador (puntos mas lejanos mas oscuros), por lo que deben ordenarse previamente los triángulos.

Ya que el objetivo del trabajo no fué diseñar un sistema completo de graficación sino demostrar la performance del enfoque dado. Se Investigó la posibilidad de incorporar rutinas de ray-tracing para obtener figuras mas realistas pero, debido a los altos costos computacionales de los algoritmos, el tiempo de respuesta era inaceptable. Existe excelente bibliografía sobre el tema, como así también paquetes preparados especialmente para visualizar datos volumétricos. Sin embargo, estos paquetes frecuentemente corren en equipos especialmente diseñados.

CAPITULO VI

DETALLES TECNICOS

§ 6.1 Equipamiento Utilizado

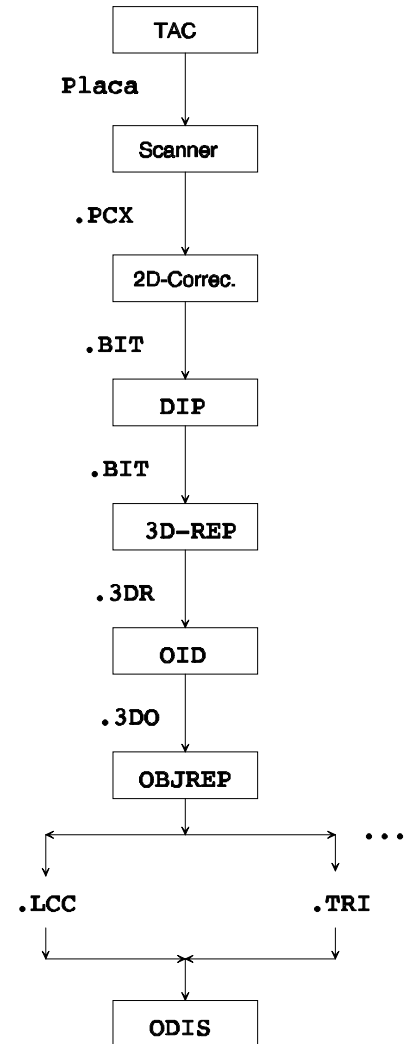
Nuestra base de implementación fué un equipo PC-486 equipado con monitor VGA de 256 colores. El equipo posee coprocesador matemático, que es esencial por la gran cantidad de cálculos de punto flotante, especialmente en el área de computer graphics y image processing.

§ 6.2 Módulos Componentes del Sistema

En la (Figura 6-1) se puede apreciar los distintos módulos componentes de un sistema típico como el realizado. Los distintos módulos son:

- TAC: El equipo de tomografía. Produce las placas con los cortes axiales.
- SCANNER: Procesa las placas y genera archivos de imágenes. Los graba en un formato estándar (por ejemplo, PCX)
- 2D-CORRECCION: Se corrigen las imágenes, anclándolas, determinando ROI/VOI. Finalmente se guardan en un formato simple (.BIT)
- DIP (Digital Image Processing): Se procesa cada imagen con distintos algoritmos para eliminar el ruido. El resultado se mantiene en (.BIT), pero ahora el ruido ha sido removido de las imágenes.
- 3DREP (3D Representation): Con el conjunto de archivos (.BIT) se forma un volumen único de datos (.3DR).
- OID (Object Identification): Se procesa el volumen de datos segmentándose el objeto de interés. El resultado queda en un archivo acorde (.3DO), por ejemplo, una escena binaria.

- OREP (Object Representation): Este módulo transforma la representación "bruta" (3DO) en diferentes estructuras de datos, por ejemplo, contornos dirigidos nivelados (.LCC), parches triangulares (.TRI) u otros.
- ODIS (Object Display): Este es el sistema de usuario final que recibe las distintas representaciones de los objetos y los muestra en pantalla.



• Fig. 6-1: Esquema de los distintos módulos de un sistema de reconocimiento de formas, generación y visualización de datos espaciales

§ 6.3 Detalles de Software

Se utilizó el sistema operativo MS-DOS en todo momento. Los lenguajes de programación usados fueron:

- Turbo Pascal 6.0: Procesamiento digital de imágenes, manejo general de archivos, detección de contornos cerrados.
- Borland C++ 3.0: Triangulación.

- Borland C 3.0: Ray Tracing.
- Turbo Pascal for Windows 1.0: Sistema de Usuario Final (graficación, shading, etc.)

El algoritmo de triangulación, conceptualmente el mas complicado, se implementó integramente en C++ utilizando técnicas de OOP, lo que facilitó mucho la tarea.

CAPITULO VII

CONCLUSIONES Y RECOMENDACIONES

§ 7.1 Conclusiones Sobre el Trabajo

Lejos de intentar producir un producto comercial, el interés básico giró siempre en torno a la resolución de un problema específico: ¿Cómo llegar del centro de diagnóstico al consultorio con la tecnología existente en nuestro país y al alcance de todos?

Para resolver este problema hemos hecho un gran esfuerzo de investigación, consultando gran cantidad de material bibliográfico.

De la experiencia obtenida podemos confirmar la tendencia general de toda la comunidad científica hacia la visualización tridimensional de los datos. Dentro de esta rama, el "volume rendering" es, sin lugar a dudas, el área de mayor impulso.

La tendencia hacia entornos GUI es absoluta, reconocida y, a nuestro parecer, totalmente justificada.

Con este trabajo no se pretende definir una metodología estandar, ni explicar diseño de sistemas, ni siquiera se intenta dar juicios definitivos sobre los métodos investigados.

Es de nuestro interés, sin embargo, presentar, explicar el tema y delinear las áreas que podrían ser profundizadas y mejoradas. Es decir, dar el puntapié inicial sobre un área donde se puede hacer, profundizar y, por sobre todo, aprender mucho.

Con el presente trabajo se ha demostrado que existe la alternativa para el low-end en cuanto a la visualización de los datos espaciales. Queda para una extensión futura la optimización de los algoritmos aquí presentados, como así también la adición de nuevos sistemas de graficación y métodos de rendering.

§ 7.2 Recomendaciones Para Futuros Trabajos

Existen muchos puntos que pueden ser implementados, mejorados y ampliados, pudiendo servir de punto de partida para futuros trabajos finales.

La lista a continuación es una recopilación de los temas que podrían extender y profundizar este trabajo.

- Investigar la posibilidad de estimar la función de ruido introducida por los aparatos de scanning y diseñar un filtro adecuado para eliminarlo.
- Diseñar y/o adquirir un sistema de adquisición de datos mas versatil que el utilizado por nosotros. Estudiar la posibilidad de integrarlo en el centro de diagnóstico y sincronizarlo con el aparato de TAC.
- Investigar algoritmos y procedimientos para lograr la registración automática (sin definir ROIs ni VOIs) de las imágenes.
- Investigar los métodos para corregir imágenes deformadas (image-warping).
- Estudiar las nuevas tendencias en la interpolación recientemente propuestas [HER92]
- Profundizar en los algoritmos de segmentación de objetos, intentando obtener reglas para la determinación de umbrales globales de manera automática, o umbrales locales ligados a parámetros cambiantes a lo largo de la imagen.
- Mejorar los algoritmos de triangulación propuestos en la bibliografía, sobre todo en lo referente a las reglas heurísticas para decidir el link entre dos contornos cualesquiera.
- Implementar la visualización por medio de superficies suaves, manteniendo los puntos de control bajo el dominio del usuario, para poder realizar deformaciones a las imágenes.
- Mejorar los algoritmos de shading/rendering, implementando inclusive foto-realismo.
- Investigar Volume Rendering. Disponiendo del hardware adecuado es el area de mayor crecimiento en el futuro cercano.

GLOSARIO

Axial, orientación: Por orientación axial de los cortes entendemos a los realizados sobre un plano transversal a la dirección de movimiento de la camilla del equipo.

Binary Scene: Espacio tridimensional de datos donde cada voxel tiene uno de dos valores posibles, según su pertenencia (o nó) al objeto que la escena binaria pretende representar.

Bit: Mínima unidad de información. Puede tomar uno de dos valores (0 o 1).

Byte: Conjunto de 8 bits. Medida utilizada normalmente para medir ciertas cantidades en computación.

Cápsula Convexa: Mínimo polígono convexo que contiene al polígono dado.

Convex Hull: ver cápsula convexa.

Digitalización: Proceso por el cual se traduce una señal de su forma analógica a una representación discreta numérica.

GUI: Graphical User Interface. Interface gráfica para el usuario.

Image Space: Sistema de coordenadas de imagen, fijadas a la pantalla.

LZW: Lempel-Ziv-Welch. Algoritmo de compresión de datos utilizado en computación.

MB: ver Megabytes.

Medical Imaging: Disciplina desarrollada a partir del nacimiento de la computación gráfica. Se ocupa de todo lo relacionado con representar partes del cuerpo mediante imágenes computacionales, utilizando fuentes de datos diversas (TAC, NMR, SPECT, Rayos-X, etc.)

Megabytes: Aproximadamente, un millón de bytes. (Exactamente, 1024×1024 bytes)

Object Space: Sistema de coordenadas universales (o del objeto). El espacio de datos original está inscripto en este sistema.

Over-Sampling: Sobre-muestreado. Se toman mas mediciones que las necesarias para un aplicación específica.

PCX: Formato de grabación de imágenes muy utilizado en el entorno PC.

Pixel: Mínimo elemento bidimensional de superficie. Representa un punto en una imagen de video.

Placa Tomográfica: Representación impresa de los resultados de uno o varios cortes obtenidos con el TAC.

Punto de Vista: Posición del observador de la imagen tridimensional.

RLL: Run-Length-Limited. Algoritmo de compresión de datos utilizado en computación.

ROI: Region de Interés. Superficie rectangular que se define de manera tal que contenga la parte de la imagen que nos interesa.

Sampling: Muestreado.

Scanner: Equipo que capta una imagen impresa y la traduce a un archivo de computadora.

Segmentación: Proceso por el cual se divide una imagen hasta identificar sólo las partes relevantes.

Shading: Sombreado, técnica para darle distintos tonos a una imagen según su ubicación, texture, posición del observador, etc.

Slice: corte realizado por el equipo TAC, según nuestra acepción.

Sub-Sampling: Sub-muestreado. Se toman menos mediciones que las necesarias.

Surface Rendering: Disciplina que intenta modelar y representar un objeto mediante el uso de entidades matemáticas.

TAC: Abreviación de Tomografía Axial Computada (utilizado también como Tomógrafo Axial Computado)

Threshold: ver umbral.

TIFF: Tagged Information File Format. Formato de grabación de imágenes muy utilizado en el entorno PC.

Umbral: Valor límite a partir del cual, una determinada función cambia de valor.

Vecindad: Se define como un entorno a un punto en el plano (vecindad bidimensional) o en el espacio (vecindad tridimensional). La manera mas común de definir la vecindad es en el plano, como los pixels adyacentes a un pixel dado (vecindad-8).

VOI: Volumen de Interés. Producto de la extrusión del máximo ROI de un conjunto de cortes, a través de toda la longitud del volumen de datos.

Volume Rendering: Disciplina que intenta modelar y representar un objeto tridimensional mediante la enumeración exhaustiva de sus elementos componentes.

Voxel: Mínimo elemento tridimensional de volumen. Representa un espacio indivisible (generalmente cúbico) dentro del espacio total de datos. Es el análogo en tres dimensiones a pixel en dos.

Wireframe: (malla de alambre). Modo de representar objetos tridimensionales, mediante sus bordes únicamente.

Workstations: Equipos especialmente diseñados para el trabajo con gráficos y / o grandes volúmenes de datos.

BIBLIOGRAFIA

- [BAR88] Barrett, W.A., Udupa, J.K.; "Dynamic Display and Quantitative Analysis of Three-Dimensional Left Ventricular Pathology"; IEEE Proceedings of Computers in Cardiology, September 1988.
- [BAR82] Barsky, B.A., Greenberg, D.P.; "Interactive Surface Representation System using a B-Spline Formulation with Interpolation Capability"; Butterworth & Co (Publishers), Vol. 14, No. 4, July 1982.
- [BEC89] Becker, S.C., Barrett, W.A.; "Interactive Morphometrics from Three-Dimensional Surface Images"; Department of Computer Science, Brigham Young University, 1989.
- [BEC91] Becker, S.C., Barrett, W.A., Olsen Jr., D.R.; "Interactive Measurement of Three-Dimensional Objects using a Depth Buffer and Linear Probe"; ACM Transactions on Graphics, Vol. 10, No. 2, April 1991.
- [BER92] Bertrand, M.; "Fast Bezier Curves in Windows"; PC-Techinques, Feb/March 1992.
- [BOI85-1] Boissonnat, J.D.; "Surface Reconstruction from Planar Cross-Sections"; 1985 IEEE Proceeding on Artificial Intelligence, 1985.
- [BOI85-2] Boissonnat, J.D., Kofakis, P.; "Use of the Delaunay Triangulation for the Identification and the Localization of Objects"; 1985 IEEE Proceedings on Artificial Intelligence, 1985.
- [CAV85] Cavendish, J.C., Field, D.A., Frey, W.H.; "An Approach to Automatic Three-Dimensional Finite Element Mesh Generation"; International Journal for Numerical Methods in Engineering, Vol. 21, 329 347, 1985.
- [CHE76] Chen, C.H.; "Pattern Recognition and Artificial Intelligence"; Academic Press, 1976.
- [DAW87] Dawson, B.M.; "Introduction to Image Processing Algorithms"; Byte Magazine, March 1987.
- [DUF79] Duff, T.; "Smoothly Shaded Renderings of Polyhedral Objects on Raster Displays"; ACM-O-89791-001-4/79/0800-270, 1979.
- [EKO91] Ekoule, A.B., Peyrin, F.C., Odet, C.L.; "A Triangulation Algorithm from Arbitrary Shaped Multiple Planar Contours"; ACM Transactions on Graphics, Vol. 10, No. 2, April 1991.

- [FOL90] Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F.; "Computer Graphics. Principles and Practice"; Addison-Wesley, 1990.
- [FRI85] Frieder, G., Herman, G.T., Meyer, C.; "Large Software Problems for Small Computers: An Example from Medical Imaging"; IEEE Software, September 1985.
- [FRI85] Frieder, G., Gordon, D., Reynolds, R.A.; "Back-to-Front Display of Voxel-Based Objects"; IEEE Computer Graphics & Applications, January 1985.
- [FU88] Fu, K.S., Gonzalez, R.C., Lee, C.S.G.; "Robótica. Control, Detección, Visión e Inteligencia"; McGraw-Hill, 1988.
- [GON87] Gonzalez, Rafael C., Wintz, Paul; "Digital Image Processing", second edition, Nov. 1987.
- [GOR91] Gordon, D., Chen, S.; "Front-to-Back Display of BSP Trees"; IEEE Computer Graphics & Applications, September 1991.
- [HER83] Herman, G.T., Udupa, J.K.; "Display of 3-D Digital Images: Computational Foundations and Medical Applications"; IEEE Computer Graphics & Applications, August 1983.
- [HER88] Herman, G.T.; "From 2D to 3D Representation"; in "Mathematics and Computer Science in Medical Imaging", NATO ASI Series, Series F, Vol. 39, 1988.
- [HER92] Herman, G.T., Zheng, J., Bucholtz, C.A.; "Shape-based Interpolation"; IEEE Computer Graphics & Applications, May 1992.
- [KER91] Kergosien, Y.L.; "Generic Sign Systems in Medical Imaging"; IEEE Computer Graphics & Applications, September 1991.
- [MAG91] Maguire Jr., G.Q. y otros; "Graphics Applied to Medical Image Registration"; IEEE Computer Graphics & Applications, March 1991.
- [MAS89] Mascarenhas, N., Velasco, F.; "Processamento Digital de Imagens", IV EBAI, Stgo. del Estero, 1989.
- [MIH91] Mihalisin, T., Timlin, J., Schwegler, J.; "Visualizing Multivariate Functions, Data, and Distributions"; IEEE Computer Graphics & Applications, May 1991.
- [NIE91] Nielson, G.M., Foley, T.A.; "Visualizing and Modeling Scattered Multivariate Data"; IEEE Computer Graphics & Applications, May 1991.
- [NOV90] Novis, K.L., Sillion, F.X., Greenberg, D.P.; "An Efficient Method for Volume Rendering using Perspective Projection"; ACM Computer Graphics, Vol. 24, No. 5, November 1990.
- [RAY90] Raya, S.P., Udupa, J.K., Barrett, W.A.; "A PC-Based 3D Imaging System: Algorithms, Software, and Hardware Considerations"; Medical Image Processing Group, Department of Radiology, University of Pennsylvania, 1990.
- [SAM90-1] Samet, H.; "The Design and Analysis of Spatial Data Structures"; Addison-Wesley, 1990.

- [SAM90-2] Samet, H.; "Applications of Spatial Data Structures"; Addison-Wesley, 1990.
- [SUT90] Suto, Y.; "Three-Dimensional Surgical Simulation System Using X-ray CT and MR Images"; Toshiba Medical Review, No. 35, 1990.
- [TAK89] Takahiro, L., "Cooperation Between Medicine and Technology", in Toshiba Medical Review No.39 1989
- [TAN80] Tanimoto, S., Klinger, A.; "Structured Computer Vision"; Academic Press, 1980.
- [TOE90] Toennies, K.D., Tronnier, U.; "3D Modeling using and Extended Cell Enumeration Representation"; ACM Computer Graphics, Vol. 24, No. 5, November 1990.
- [UDU90] Udupa, J.K., Raya, S.P., Barrett, W.A.; "A PC-Based 3D Imaging System for Biomedical Data"; Proceedings of the First Conference on Visualization in Biomedical Computing, May 1990.
- [WAN85] Wang, Y.F., Aggarwal, J.K.; "Construction of Surface Representation from 3-D Volumetric Scene Description"; IEEE Proceeding in Artificial Intelligence, 1985.
- [YER84] Yerry, M.A., Shephard, M.S.; "Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique"; International Journal for Numerical Methods in Engineering, Vol. 20, 1965-1990, 1984.

APENDICE

Los distintos módulos que componen este sistema son:

- PCX2BIT.PAS: Procesa los archivos .PCX resultado del proceso de scaneo. Registra las imágenes, filtra y genera los .BIT
- BIT2CON.PAS: Para un juego completo de archivos .BIT, identifica los contornos cerrados y genera múltiples archivos .CON (1 por cada .BIT)
- CON2DAV.PAS: Procesa cada archivo .CON, rectificando y ordenando el contorno cerrado.
- DAV2LCC.PAS: Concentra un juego de archivos .DAV en un solo archivo de .LCC (Leveled Closed Contours)
- LCC2TLI.CPP: Algoritmo de triangulación. Genera una lista simple de triángulos.
- TLI2TRI.CPP: Procesa la lista de triángulos para optimizar el espacio y mejorar el display.
- 3DEYES.PAS: End-User System. Lee los .LCC y .TRI y los grafica (bajo MS-Windows).
- CATMRT.PAS: Sub-Sistema de procesamiento digital de imágenes. Lee cualquier archivo .PCX y aplica filtros a elección del usuario.

Se adjunta soporte magnético con los distintos módulos arriba expuestos.

Este trabajo se terminó de editar el 13 de Septiembre de 1992.

El texto se escribió en WordPerfect 5.1, Windows Write, Microsoft Word for Windows 2.0 y Ami Pro for Windows 2.0. La edición se realizó en Ami Pro for Windows 2.0. Los gráficos se obtuvieron de pantallas del sistema, bajo Microsoft Windows, utilizando el clipboard, se editaron en Micrografx Designer 3.0 y luego se importaron en Ami Pro.

(c) - Luis Ricardo Blando y Diego Antonio Viñas - 1992

Re-editado el 16 de Octubre de 1995 por Luis Blando.